

스튜디오 안내서

AnyLink 7

TMAXSOFT

저작권 공지

Copyright 2025. TmaxSoft Co., Ltd. All Rights Reserved.

회사 정보

(주)티맥스소프트

주소 : 경기도 성남시 분당구 황새울로258번길 29, 티맥스수내타워 8-9층

기술 서비스 센터: 1544-8629

홈페이지: <https://www.tmaxsoft.com>

제한된 권리

이 소프트웨어(Tmax AnyLink®) 사용설명서와 프로그램은 저작권법과 국제 조약에 의해 보호됩니다. 사용설명서와 프로그램은 TmaxSoft Co., Ltd.와의 사용권 계약 하에서만 사용할 수 있으며, 사용설명서는 사용권 계약의 범위 내에서만 배포 또는 복제할 수 있습니다. 이 사용설명서의 전부 또는 일부를 TmaxSoft의 사전 서면 동의 없이 전자, 기계, 녹음 등의 수단으로 전송, 복제, 배포하거나 2차적 저작물을 작성할 수 없습니다.

이 소프트웨어 사용설명서와 프로그램의 사용권 계약은 어떠한 경우에도 사용설명서 및 프로그램과 관련된 지적 재산권(등록 여부를 불문)을 양도하는 것으로 해석되지 않으며, 브랜드나 로고, 상표 등을 사용할 권한을 부여하지 않습니다. 사용설명서는 오로지 정보 제공만을 목적으로 하며, 이로 인한 계약상의 직접적 또는 간접적 책임을 지지 않습니다. 또한 사용설명서 상의 내용이 법적 또는 상업적인 특정 조건을 만족시킬 것을 보장하지 않습니다. 사용설명서는 제품의 업그레이드나 수정에 따라 예고 없이 변경될 수 있으며, 내용상의 오류가 없음을 보장하지 않습니다.

상표 공지

Tmax AnyLink®는 TmaxSoft Co., Ltd.의 등록 상표입니다. 본 사용설명서에 기재된 모든 제품과 회사 이름은 각각 해당 소유주의 상표로서 참조용으로만 사용되며 반드시 상표 표시 (™, ®)를 하지는 않습니다.

오픈소스 소프트웨어 공지

본 제품의 일부 파일 또는 모듈은 다음의 라이선스를 준수합니다. : APACHE2.0, CDDL1.0, EDL1.0, OPEN SYMPHONY SOFTWARE1.1, TRILEAD-SSH2, Bouncy Castle, BSD, MIT, SIL OPEN FONT1.1

관련 상세 정보는 제품의 다음 디렉터리에 기재된 사항을 참고하시기 바랍니다. : \${AnyLink_HOME}\AnyLink-licenses

유지 보수

구분	지원항목	서비스 내용
제품지원	패치 & 업그레이드	무상 패치 서비스 제공 메이저 버전 업그레이드 시 할인 혜택 웹 지원을 통한 패치 내역 제공
기술 지원 - 기본 서비스	장애 지원	장애 발생 시 원인 분석 및 조치 Service Desk팀 → 기술팀 → R&D의 3단계 장애 분석 및 조치
	일상 지원(온라인 지원)	E-mail, 전화, 원격, 웹 사이트 등 온라인 자원을 통한 질의 응답 서비스
	고객 맞춤 지원(방문 지원)	고객의 요청으로 수행하는 방문 지원 서비스
기술 지원 - 옵션 서비스	예방 지원	정기 점검을 통한 시스템 운영현황 보고 및 장애 예방 <ul style="list-style-type: none"> 관리자 또는 운영자의 요구사항 수렴 운영 현황(시스템, 엔진 운영) 보고서 제공 필요 시 시스템 개선 권장 사항 보고
유지 보수 비용 및 기간	계약 시 별도 협의	계약 시 EOL/EOS 문서 제공

안내서 이력

제품 버전	안내서 버전	발행일	비고
AnyLink 7	3.1.2	2025-01-20	-
AnyLink 7	3.1.1	2023-03-13	-
AnyLink 7	2.1.4	2019-08-23	-
AnyLink 7	2.1.1	2017-03-24	-

목차

1. 소개	1
1.1. 프로그램 시작	1
1.2. 화면구성	4
1.2.1. Menu	6
1.2.2. View	12
1.2.3. 네비게이터	13
1.3. 리소스	16
1.4. 프로젝트 생성	17
2. 거래/거래그룹 에디터	19
2.1. 거래/거래그룹 생성	19
2.2. 거래/거래그룹 에디터	21
2.2.1. 거래/거래그룹 정보	21
2.2.2. 거래/거래그룹 옵션	31
2.2.3. 파싱정보	34
2.2.4. 파싱 옵션	36
2.2.5. XML 뷰어	38
3. 메시지 에디터	39
3.1. 메시지 생성	39
3.2. 메시지 에디터	40
3.2.1. 메시지 설정	41
3.2.2. 메시지 구성	45
4. 아웃바운드 룰 에디터	54
4.1. 개요	54
4.2. 커스텀 로그 아웃바운드 룰	54
4.2.1. 커스텀 로그 아웃바운드 룰 생성	54
4.2.2. 커스텀 로그 아웃바운드 룰 설정	55
4.2.3. 커스텀 로그 예제	60
4.3. 배치 아웃바운드 룰	65
4.3.1. 배치 아웃바운드 룰 생성	65
4.3.2. 배치 아웃바운드 룰 설정	68
5. 멀티바인딩 룰 에디터	88
5.1. 멀티바인딩 룰	88
5.2. 멀티바인딩 룰 생성	88
5.3. 멀티바인딩 설정	89
5.4. 매핑 설정	91
5.4.1. 매핑 추가/삭제	92
6. 서비스 플로우 에디터	94
6.1. 개요	94
6.2. 서비스 플로우 에디터 생성	94

6.3. 서비스 플로우 다이어그램	95
6.4. 액티비티	98
6.4.1. 액티비티 설정	100
6.4.2. 특정 액티비티 설정	114
6.5. 이벤트	124
6.5.1. 이벤트 설정	124
6.5.2. 타입별 이벤트 설정	125
6.6. 게이트웨이	128
6.7. 스웜레인 / 블록 / ANNOTATIONS	129
6.7.1. 블록 액티비티	130
6.8. Utility	131
6.8.1. SESSION CLOSE ACTIVITY	131
6.8.2. NETWORK MANAGEMENT ACTIVITY	132
6.8.3. MANAGER APPROVAL ACTIVITY	134
6.8.4. HTTP MULTIPART ACTIVITY	136
6.8.5. BATCH PROGRESS ACTIVITY	137
6.8.6. TCP MESSAGE SPLIT ACTIVITY	139
6.8.7. CONTROL SESSION ACTIVITY	141
7. 유저 클래스/핸들러 에디터	143
7.1. 유저 클래스	143
7.1.1. 유저 클래스 생성	143
7.1.2. 유저 클래스 정의	143
7.1.3. getEnv 함수	144
7.2. 핸들러	145
7.2.1. 핸들러 생성	145
7.2.2. 핸들러 정의	146
8. 공통업무	160
8.1. 공통업무 프로젝트 생성	160
8.2. 공통업무 거래(그룹) 생성	161
9. 배포 및 배포해제	164
9.1. 개요	164
9.2. 리소스 배포(재배포)	164
9.2.1. 배포 단위	164
9.2.2. 배포 과정	164
9.3. 리소스 배포해제	166
9.3.1. 배포해제 단위	167
9.3.2. 배포해제 과정	167
9.4. 공통 라이브러리 배포 및 배포해제	168
9.4.1. 공통 라이브러리 배포	168
9.4.2. 공통 라이브러리 배포해제	169
9.4.3. 라이브러리 추가	170

9.4.4. 라이브러리 다운로드	170
10. 리소스 및 다운로드	172
10.1. 개요	172
10.2. 리소스 검색 및 다운로드	172
부록 A: 다이얼로그	174
A.1. 메시지 관련 다이얼로그	174
A.1.1. 메시지 선택 다이얼로그	174
A.1.2. 메시지 필드 설정 다이얼로그	174
A.2. 매핑 관련 다이얼로그	175
A.2.1. 매핑 마법사	175
A.2.2. 매핑 마법사 사용	179
A.3. 리소스 관련 다이얼로그	186
A.3.1. 리소스 생성 다이얼로그	186
A.3.2. 리소스 검색 다이얼로그	187


1. 소개

본 장에서는 스튜디오의 기본 화면구성에 대해서 설명한다.

1.1. 프로그램 시작

AnyLink 스튜디오는 AnyLink 제품군 중 하나로 플로우 서비스와 각종 어댑터들의 룰을 제작하고, 디플로이(Deploy)를 편리하게 할 수 있도록 도와주는 툴이다. 사용자는 GUI 환경의 AnyLink 스튜디오를 이용하여 플로우 또는 룰을 쉽게 편집하고 관리할 수 있다. 플로우 서비스 및 각종 어댑터에 대한 설명은 해당 안내서를 참고한다.

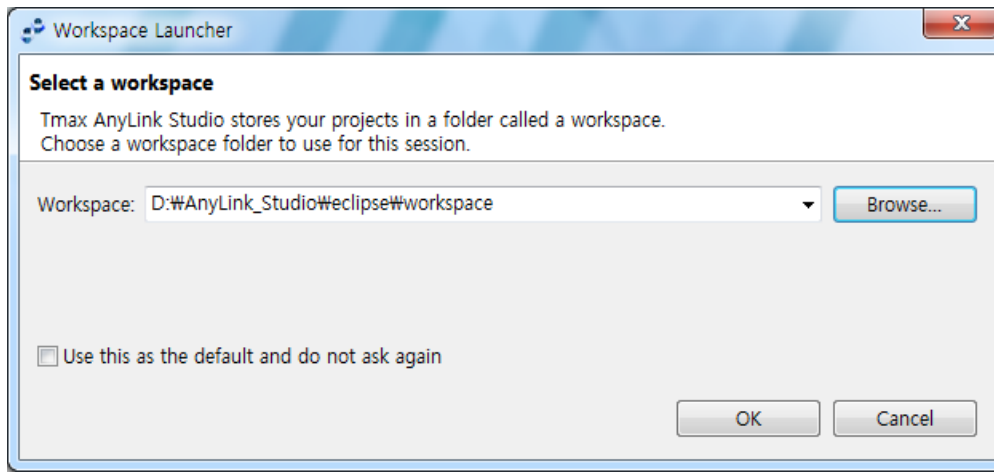
다음은 AnyLink 스튜디오를 시작하는 과정에 대한 설명이다.

1. AnyLink 스튜디오 프로그램을 시작하기 위해 **바탕화면 아이콘**( AnyLink.exe)을 클릭한다.
2. 스튜디오를 로딩하는 동안 AnyLink 로고, 저작 문구 및 진행 상태를 보여주는 화면이 나타난다.



스튜디오 실행 화면

3. 스튜디오를 실행하면 Workspace Launcher가 실행되고, Workspace를 설정할 수 있다. **[OK]** 버튼을 클릭하면 AnyLink 스튜디오가 실행된다.



Workspace Launcher

4. 스튜디오 화면의 상단 툴바의  버튼을 클릭하면 **로그인 화면**으로 이동한다.

스튜디오는 각 사용자의 변경이력을 관리하기 위해 로그인을 필요로 한다. 로그인한 사용자가 서버에 디플로이한 정보는 변경이력으로 관리되며 사용자 간의 데이터 충돌을 감지하는 데 사용한다.

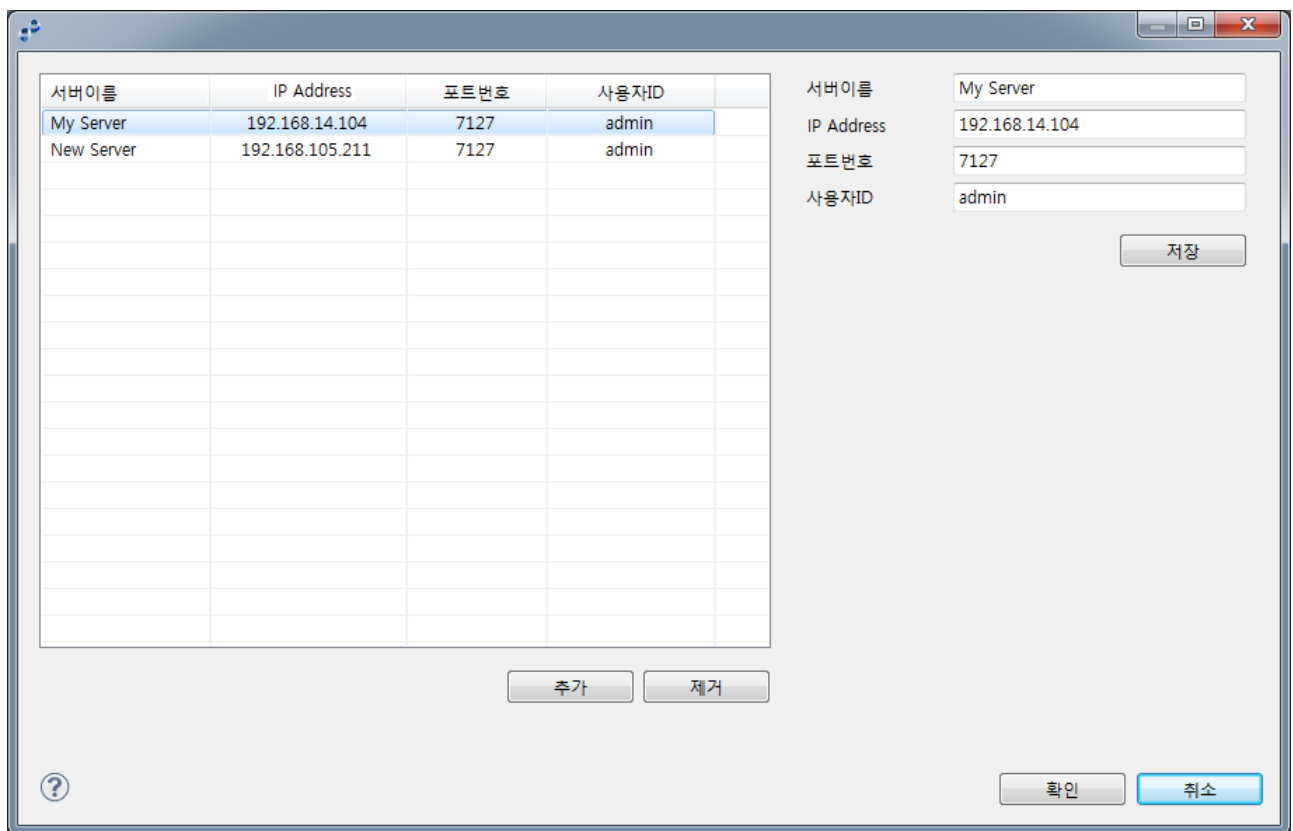
로그인 화면에서 서버 이름, 사용자 아이디, 패스워드, 서버 IP 주소 및 포트번호를 입력하고 **[로그인]** 버튼을 클릭하여 AnyLink 서버에 접속한다.



로그인 화면

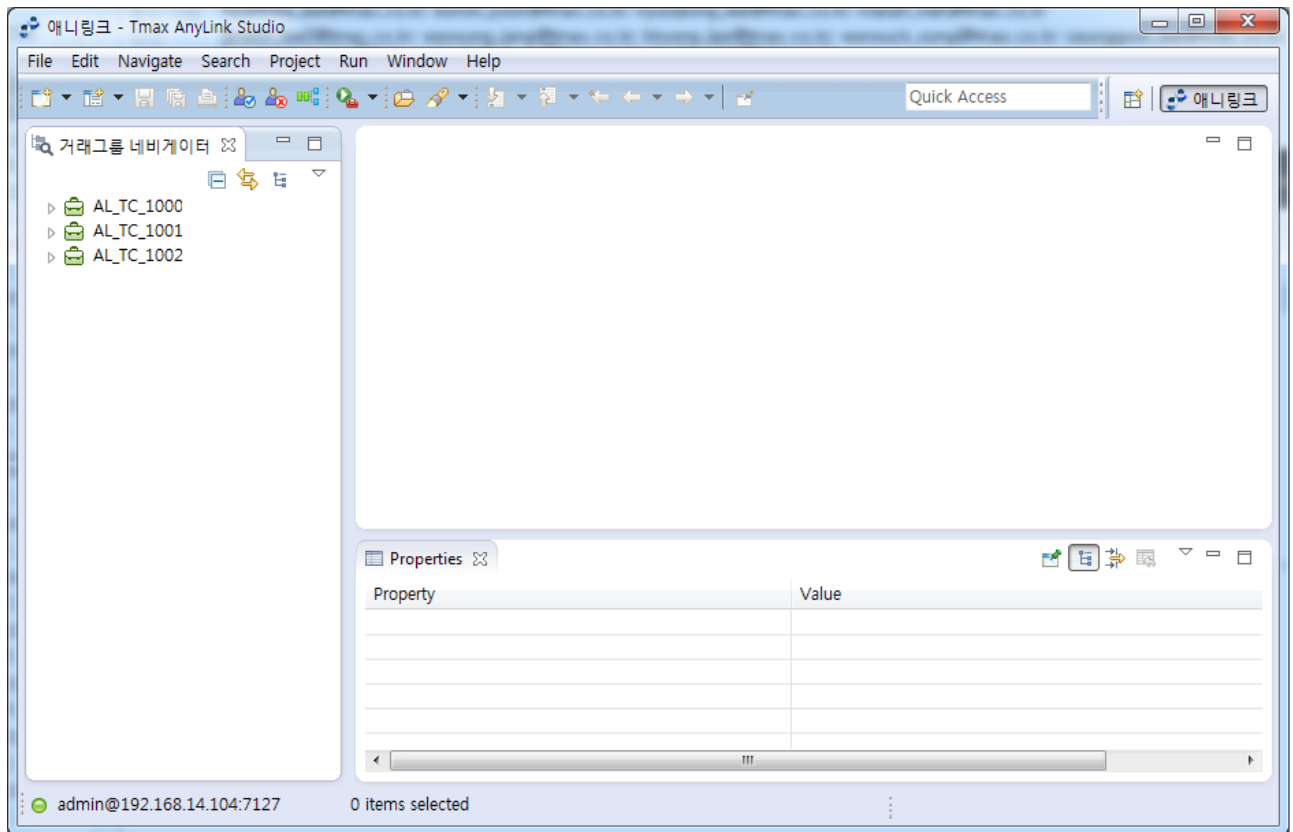
항목	설명
서버 이름	로그인 정보(주소, 포트, 사용자 아이디)가 저장된다. 서버 이름에는 로그인 아이콘을 통해 서버와 접속하지 않으면, 자동으로 로컬 모드에서 스튜디오를 사용할 수 있다.
IP Address	DIS가 설치되어 있는 서버의 IP 주소이다.
포트번호	최초 설치할 때 7127로 설정되어 있다.
사용자 ID	WebAdmin에서 사용자로 등록된 아이디이다.
패스워드	WebAdmin에서 설정된 패스워드이다.

로그인 화면에서 [편집] 버튼을 클릭하면 로그인 정보를 추가/편집/삭제할 수 있는 다이얼로그가 생성된다.



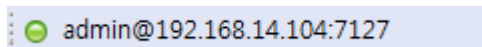
로그인 정보 편집 다이얼로그

5. 로그인에 성공하면 메인 화면이 나타난다.



스튜디오 메인 화면

화면 아래에 다음과 같이 로그인 접속 정보가 표시된다.

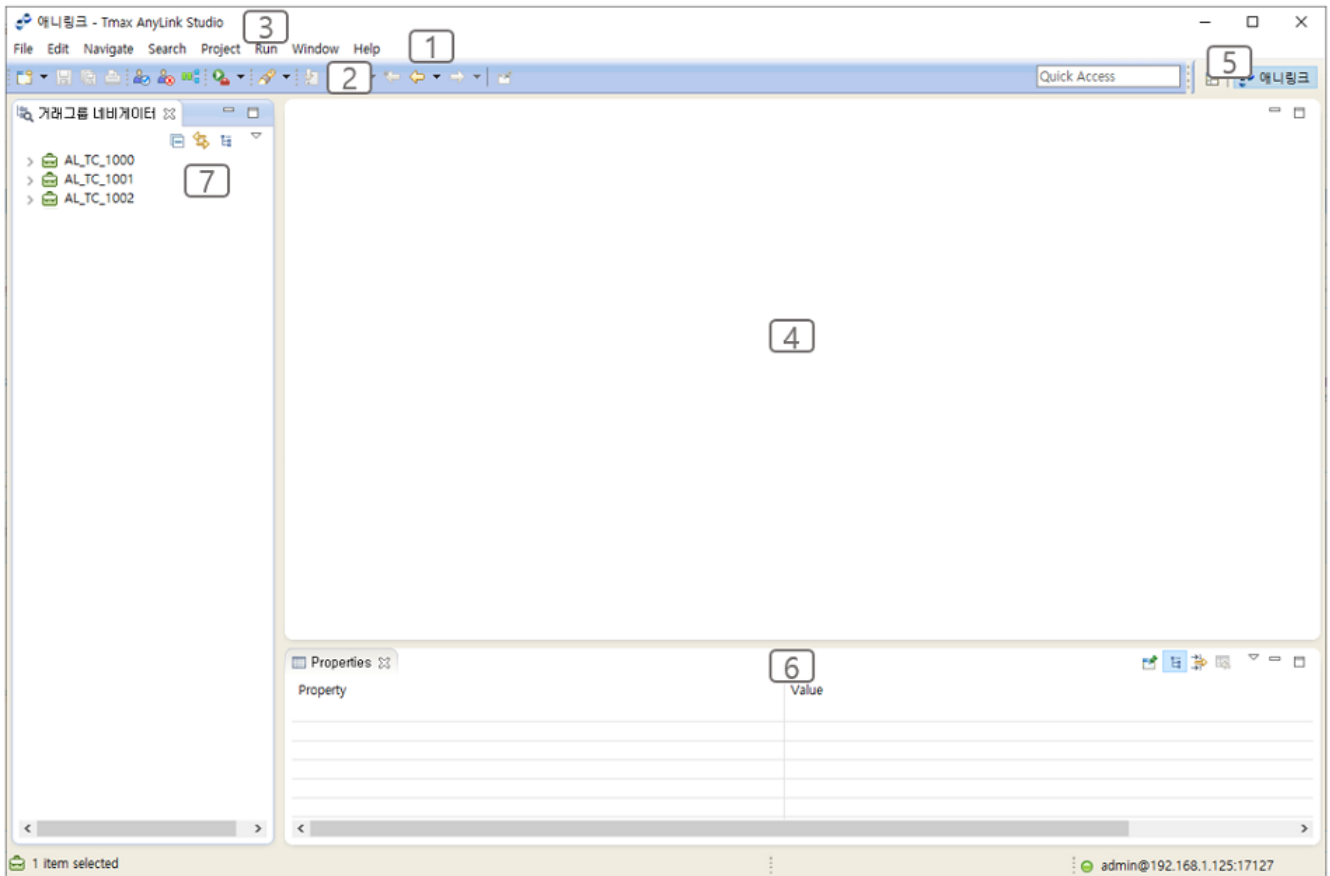


스튜디오 메인 화면 - 로그인 정보

1.2. 화면구성

스튜디오는 하나의 큰 화면인 Workbench와 그 위에 여러 개의 Perspective(레이아웃)으로 구성된다.

다음은 AnyLink 스튜디오 초기화면이다.



스튜디오 초기화면



• (1) Menu

Workbench의 메뉴를 보여준다. 각 메뉴에 대한 자세한 설명은 [Menu](#)를 참고한다.

• (2) Tool bar

빠른 메뉴 실행을 위해 메뉴에 있는 명령 중 일부를 툴바에 표시한다. 툴바에 표시되는 메뉴 목록은 **[Window] > [Customize Perspective]**에서 설정한다.

다음은 AnyLink에서 제공하는 툴바이다.

아이콘	설명
 (로그인)	DIS에 접속한다.
 (로그아웃)	DIS 접속을 해제한다.
 (관계도)	리소스의 연관 관계를 조회할수 있는 View를 생성한다.

• (3) Title bar

다음의 형식으로 Workbench의 타이틀이 표시된다.

{현재 Perspective 이름} - {현재 작업중인 파일명} - Tmax AnyLink Studio

• (4) Editor Panel

서비스 플로우 다이어그램 편집기 등의 편집기가 위치한다.

- (5) Perspective

Perspective의 단축 버튼이 나타나며, 현재 활성화된 Perspective는 밝게 표시된다. 마우스 오른쪽 버튼을 클릭하면 나타나는 서브 메뉴에서 톨바에 표시되는 버튼에 대한 설정을 할 수 있다.

- (6) View

리소스를 탐색하고 리소스의 프로퍼티를 수정할 때 사용한다. 각 메뉴에 대한 자세한 설명은 [View](#)를 참고한다.

- (7) 네비게이터

편집기를 지원하며, 대체로 Presentation 또는 Workbench의 정보 탐색을 제공한다. 자세한 설명은 [네비게이터](#)를 참고한다.

1.2.1. Menu

Workbench는 다음의 메뉴로 구성된다.

- [File] 메뉴

파일 생성, 저장에 관련한 메뉴이다.

메뉴	설명
[New]	프로젝트 또는 파일을 생성한다.
[Open File]	일반적으로 이클립스(eclipse)에서 편집기를 여는 방법은 Service Explorer 에서 Workspace 안에 있는 파일을 마우스로 더블클릭을 하거나, Service Explorer 에서 파일을 선택한 후 컨텍스트 메뉴에서 [Open File] 을 선택한다. 명령을 사용하면 Workspace 내에 있지 않은 파일을 열수 있다.
[Close]	현재 활성화된 편집기를 닫는다.
[Close All]	현재 열려 있는 모든 편집기를 닫는다.
[Save]	현재 활성화된 편집기 파일을 저장한다.
[Save As]	현재 활성화되어 있는 편집기 파일을 본래 이름이 아닌 다른 이름으로 변경하여 저장한다.
[Save All]	현재 열려있는 모든 편집기의 파일들을 저장한다.
[Revert]	현재 활성화된 편집기 파일을 마지막 저장한 상태로 되돌린다. 마지막 저장한 이후에 편집기에서 변경된 내용은 적용되지 않는다.
[Move]	Service Explorer 에서 선택한 파일 또는 폴더를 마우스로 끌어다가 다른 위치에 놓음으로써, 동일한 동작을 편리하게 수행할 수 있다.
[Rename]	Service Explorer 에서 선택한 파일 또는 폴더의 이름을 변경한다.

메뉴	설명
[Refresh]	<p>Service Explorer에서 보여주는 Workspace의 내용을 실제 파일 시스템의 내용으로 새로고침한다.</p> <p>Workspace 내의 프로젝트는 실제 파일 시스템과 동일한 내용을 보여준다. Workbench 외부적으로 변경/추가/삭제된 파일에 대해서는 Workbench는 알지 못한다. 이러한 경우에 이 Refresh 명령을 사용함으로써 Workbench와 실제의 파일 시스템과 동기를 맞춘다.</p>
[Convert Line Delimiters To]	현재 활성화되어 있는 편집기의 라인 변경 구분자를 Window/Unix/MacOS9의 라인 변경자로 변경시킨다. 현재 Window가 디폴트로 설정되어 있다.
[Print]	현재 활성화되어 있는 편집기의 내용을 인쇄한다.
[Switch Workspace]	새 Workspace를 생성과 동시에 전환하거나 다른 Workspace로 전환한다.
[Restart]	스튜디오를 재시작하는 명령이다.
[Import]	Wizard를 사용하여 외부의 프로젝트 또는 파일을 Workspace 내부로 가져온다.
[Export]	Wizard를 사용하여 Workspace 내의 프로젝트 또는 파일을 외부로 내보내기(export)하는 명령이다. Wizard에서는 다양한 형태의 내보내기 방법을 제공한다.
[Properties]	Service Explorer 에서 선택한 리소스에 대해 속성 대화상자를 보여준다.
[Recent Files]	<p>Workbench에서 가장 최근에 작업한 파일의 목록을 보여준다.</p> <p>[Recent Files]에 표시되는 파일의 최대 개수는 [Windows] > [Preference] > [General] > [Editors] 메뉴에서 'Size of recently opened files list' 항목을 변경한다.</p>
[Exit]	Workbench를 종료한다. Workbench를 종료하면 종료 시점의 Perspective와 열려있는 편집기 정보를 자동으로 저장하며, 다음에 Workbench를 열면 마지막에 저장된 Workbench 상태를 그대로 보여준다.

• [Edit] 메뉴

편집기에서 파일을 편집할 때 사용하는 메뉴이다.

메뉴	설명
[Undo]	마지막으로 편집한 내용을 취소한다.
[Redo]	Undo 명령을 사용하여 마지막으로 취소한 내용을 다시 적용한다.
[Cut]	선택한 내용을 클립보드로 복사하고, 원본을 삭제한다.
[Copy]	선택한 내용을 클립보드로 복사한다.
[Paste]	클립보드에 있는 텍스트나 객체를 활성화되어 있는 편집기 또는 뷰의 커서 위치에 삽입한다.
[Delete]	선택한 내용을 삭제한다.

메뉴	설명
[Select All]	현재 활성화되어 있는 편집기 또는 뷰의 모든 텍스트나 객체를 선택한다.
[Find/Replace]	현재 활성화되어 있는 편집기에서 표현식을 검색하고 선택적으로 표현식을 새 표현식으로 변경한다.
[Add Bookmark]	<p>활성화되어 있는 편집기에서 커서가 있는 행에 책갈피를 추가한다.</p> <p>현재 표시되어 있는 책갈피의 리스트는 [Window] > [Show View] > [Other] 메뉴를 선택하여 나타나는 Show View 대화상자의 [General] > [Bookmark]에서 조회할 수 있다.</p>
[Add Task]	<p>활성화되어 있는 편집기에서 커서가 있는 행에 태스크를 추가한다.</p> <p>현재 표시되어 있는 태스크의 리스트는 [Window] > [Show View] > [Other] 메뉴를 선택하여 나타나는 Show View 대화상자의 [General] > [Tasks]에서 조회할 수 있다.</p>

• [Navigate] 메뉴

Workbench 상의 리소스 및 항목들을 찾거나 탐색하는 메뉴이다.

메뉴	설명
[Go Into]	<p>현재 활성화되어 있는 뷰가 Tree View일 경우, 현재 선택된 아이템이 트리의 루트가 되도록 한다.</p> <p>트리의 루트는 트리에 나타나지 않는 Invisible Root를 말한다.</p>
[Go To]	<ul style="list-style-type: none"> • Back : HTML 브라우저의 [이전] 또는 [뒤로] 버튼과 유사하다. 활성화되어 있는 뷰를 Go Into 명령이 실행되기 이전 모습으로 되돌린다. • Forward : HTML 브라우저의 [다음] 또는 [앞으로] 버튼과 유사하다. 활성화되어 있는 뷰를 Go To의 Back 명령이 실행되기 이전의 모습으로 되돌린다. • Up One Level : 현재 노드의 루트를 부모 루트로 하여, 트리에서 보이는 레벨을 한 수준 올린다.
[Open from Clipboard]	클립 보드로부터 저장된 항목을 open한다.
[Show In]	다른 뷰에서 현재 활성화되어 있는 뷰의 콘텐츠를 볼 수 있도록 한다.
[Next]	<p>현재 활성화되어 있는 뷰에서 목록 또는 테이블의 다음 항목을 탐색한다.</p> <p>예를 들어 Search View가 활성화되면 다음의 검색 항목을 탐색한다.</p>
[Previous]	<p>현재 활성화되어 있는 뷰에서 목록 또는 테이블의 이전 항목을 탐색한다.</p> <p>예를 들어 Search View가 활성화되면 이전의 검색 항목을 탐색한다.</p>
[Last Edit Location]	마지막으로 편집한 위치로 이동한다.
[Back]	편집기에서 이전 위치로 이동한다.

메뉴	설명
[Forward]	편집기에서 back 명령을 취소하기 위하여 사용한다. back 명령이 수행되기 이전의 위로 이동한다.

- [Search] 메뉴

Workbench에서 파일 또는 텍스트를 검색할 수 있는 메뉴이다.

메뉴	설명
[Search]	파일과 Java 소스 내에서 검색한다.
[File]	파일과 Java 소스 내에서 검색한다.
[Text]	선택한 내용을 Workspace 내에서 검색한다.

- [Project] 메뉴

Workspace 내의 프로젝트에 관련한 메뉴이다.

메뉴	설명
[Open Project]	현재 선택한 프로젝트를 연다. 이 명령을 수행하기 위해서는 선택한 프로젝트가 닫혀 있어야 한다.
[Close Project]	현재 선택한 프로젝트를 닫는다. 이 명령을 수행하기 위해서는 선택한 프로젝트가 열려 있어야 한다.
[Build All]	Workbench의 모든 프로젝트에 대해 증분 빌드(Incremental Build)를 수행한다. 즉, 마지막으로 증분 빌드를 수행한 후 Workbench에 변경된 리소스가 있으면 해당 리소스를 모두 빌드(컴파일)한다. 이 명령은 [Project] > [Build Automatically] 메뉴를 체크하지 않은 경우에만 사용이 가능하다.
[Build Project]	현재 선택한 프로젝트에 대해 증분 빌드를 수행한다. 즉, 마지막으로 빌드를 수행한 후 변경된 리소스가 있으면 해당 리소스를 모두 빌드한다. 이 명령은 [Project] > [Build Automatically] 메뉴를 체크하지 않은 경우에만 사용이 가능하다.
[Build Working Set]	사용자가 지정한 프로젝트(working set)에 대하여 증분 빌드를 수행한다. 즉, 마지막으로 빌드한 후 리소스 변경사항에 의해 영향을 받는 working set에 있는 모든 리소스를 빌드한다. 이 명령은 [Project] > [Build Automatically] 메뉴를 체크하지 않은 경우에만 사용이 가능하다.

메뉴	설명
[Clean]	이전 빌드 결과를 모두 삭제한다. [Project] > [Build Automatically] 메뉴가 체크되어 있으면, 기본적으로 현재 프로젝트에 대해 다시 빌드를 수행한다.
[Build Automatically]	이 메뉴 항목에 체크되어 있으면, Workbench에서 리소스가 변경되어 저장에 필요할 경우에 자동으로 증분 빌드를 수행한다.
[Properties]	선택한 프로젝트 또는 선택한 리소스를 포함하는 프로젝트의 속성을 표시하는 대화상자를 연다.

• [Run] 메뉴

컴파일을 실행하거나 컴파일에 사용되는 도구 등을 설정한다.

메뉴	설명
[Run As]	설정된 컴파일 툴로 컴파일을 수행한다.
[External Tools Configurations]	컴파일 등을 수행할 외부 툴을 설정한다.
[Organize Favorites]	설정된 외부 툴을 즐겨찾기에 등록하여 [Run As] 메뉴에서 쉽게 사용하도록 한다.

• [Window] 메뉴

Workbench 화면에서 Perspective 또는 뷰 등을 열고 닫거나 설정하는 메뉴이다.

메뉴	설명
[New Window]	현재 Perspective와 동일한 Perspective가 있는 새로운 Workbench 화면을 연다.
[New Editor]	현재 선택한 파일을 새로운 편집기에서 연다.
[Hide Toolbar]	상단에 위치한 툴바를 화면에서 숨긴다.
[Open Perspective]	원하는 Perspective를 선택하여 연다.
[Show View]	원하는 뷰를 선택하여 화면에 띄운다. AnyLink 스튜디오에서 작업할 때 필수적인 뷰만 서브 메뉴에 나타나며, 이외의 다른 뷰를 조회하려면 [Other] 서브 메뉴에서 다른 뷰를 선택한다.
[Customize Perspective]	현재 Perspective에서 나타나는 메뉴, 툴바 명령 등을 설정한다.
[Save Perspective As]	기존 Perspective를 재설정하거나 현재의 Perspective를 다른 이름으로 저장한다.
[Reset Perspective]	현재의 Perspective를 원래의 레이아웃으로 복원한다.
[Close Perspective]	현재 활성화된 Perspective를 닫는다.
[Close All Perspectives]	Workbench에 열려 있는 모든 Perspective를 닫는다.

메뉴	설명
[Navigation]	<p>Workbench 화면에서 뷰, 편집기, Perspective를 이동하기 위한 메뉴이다. 이 메뉴에 표시된 단축 키를 이용하면 마우스를 사용하지 않고 키보드 만으로도 Workbench 안의 화면을 탐색할 수 있다.</p> <p>다음은 서브 메뉴에 대한 설명이다.</p> <ul style="list-style-type: none"> • [Show System Menu] : 현재 뷰 또는 편집기 크기를 조정하거나 닫을 수 있는 시스템 메뉴를 보여준다. • [Show View Menu] : 현재 활성화되어 있는 뷰에서 리스트 메뉴를 보여준다. • [Quick Access] : 문자열을 입력하여 리소스를 빠르게 검색할 수 있다. • [Maximize / Minimize Active view or Editor] : 현재의 뷰 또는 편집기를 전체 화면으로 하거나, 전체 화면인 경우에는 원래대로 되돌린다. • [Activate Editor] : 현재의 편집기를 활성화시킨다. • [Next Editor] : 가장 최근에 사용한 편집기 목록에서 다음 편집기로 이동한다. • [Previous Editor] : 가장 최근에 사용한 편집기 목록에서 이전의 편집기로 이동한다. • [Switch to Editor] : 현재 열려있는 편집기 목록을 보여주며, 선택한 편집기로 전환할 수 있다. 편집기 목록에서 편집기를 선택하면 해당 편집기가 활성화된다. • [Next View] : 가장 최근에 사용한 뷰 목록에서 다음 뷰로 이동한다. • [Previous View] : 가장 최근에 사용한 뷰 목록에서 이전 뷰로 이동한다. • [Next Perspective] : 가장 최근에 사용한 Perspective 목록에서 다음의 Perspective로 이동한다. • [Previous Perspective] : 가장 최근에 사용한 Perspective 목록에서 이전의 Perspective로 이동한다.
[Preferences]	<p>Workbench 또는 설치되어 있는 모든 플러그인에 대한 환경을 설정하는 화면이 나타난다.</p> <p>스튜디오 리소스는 XML 형식으로 생성되므로 [General] > [Workspace] 화면의 'Text file encoding' 항목은 'UTF-8'로 설정한다.</p>

• [Help] 메뉴

도움말, 제품 정보를 볼 수 있는 메뉴이다.

메뉴	설명
[Help Contents]	도움말 목록 및 상세 도움말을 제공한다.

메뉴	설명
[Search]	도움말 내용을 검색할 수 있는 검색 화면 열린다.
[Dynamic Help]	관련 토픽별로 도움말을 검색할 수 있는 화면이 열린다.
[Key Assist]	각 메뉴 항목의 단축 키에 대한 정보를 제공한다.
[Cheat Sheets]	설치된 소프트웨어에 대한 Cheat Sheets를 선택할 수 있는 화면을 제공한다.
[Check for Updates]	설치된 소프트웨어 목록과 업데이트 상태를 보여준다.
[Install New Software]	검색을 통해 소프트웨어 목록을 보여주고, 설치할 수 있는 화면을 제공한다.
[About Tamx AnyLink Studio]	AnyLink 스튜디오의 제품 정보를 제공한다.

1.2.2. View

편집기를 지원하며, 대체로 Presentation 또는 Workbench의 정보 탐색을 제공한다. 뷰는 단독으로 나타날 수도 있고 탭이 달린 형태(노트북)로 다른 뷰와 함께 나타날 수도 있다. Workbench는 탭이 화면의 상단 또는 하단에 있는지 여부를 포함하여 편리하고 신속한 여러 환경 구성 방법을 제공한다.

탭이 달린 노트북의 일부인 뷰를 활성화하려면 해당 탭을 클릭한다.

다음은 각 탭에 대한 설명이다.

- **[Project Explorer]**

Workspace 내의 리스트를 보여주며, 특정 리소스를 선택하여 편집기를 여는 등의 동작을 수행한다.

- **[Console View]**

스튜디오 사용의 여러 정보(디플로이 과정, 에러 메시지 등)를 조회한다.

- **[Error Log View]**

Workbench 수행 중 에러가 발생한 경우에 로그를 조회한다.

- **[Outline View]**

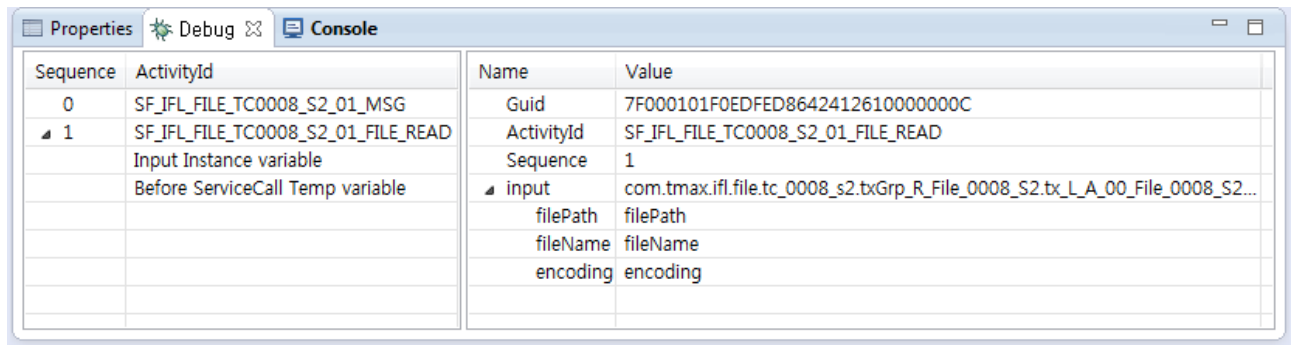
활성화되어 있는 편집기의 내용에 대한 개요를 조회한다.

- **[Properties View]**

Workbench에서 선택된 객체의 상세 정보를 보여주고 수정할 수 있다.

- **[Debug View]**

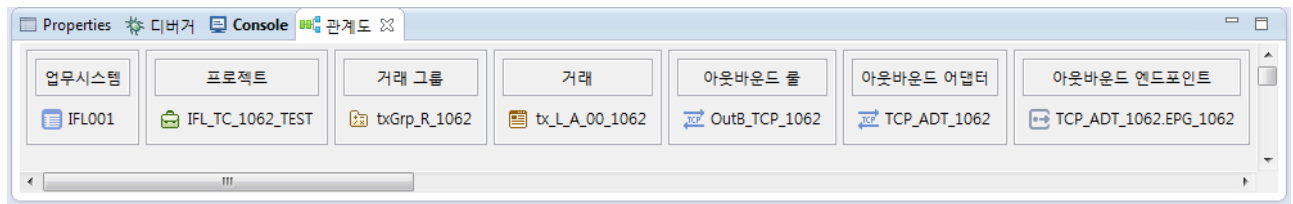
플로우 디버거 결과를 조회한다.



디버거 View 화면

• [관계도 View]

리소스 간의 연관 관계를 조회한다.

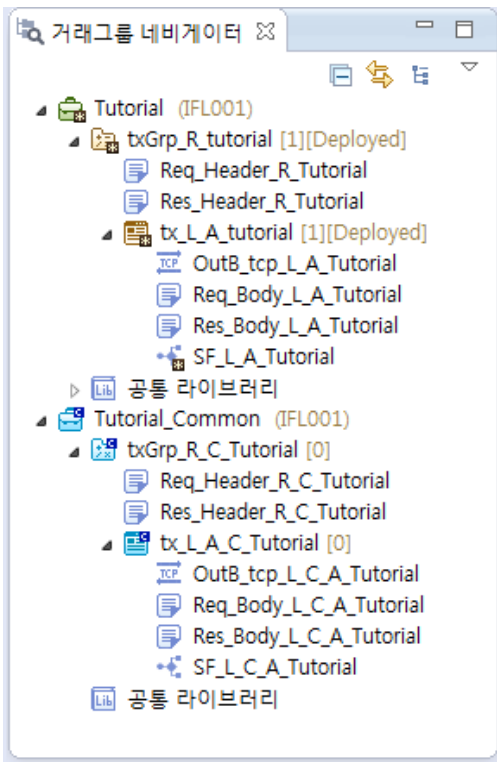


관계도 View 화면

1.2.3. 네비게이터

네비게이터는 단독으로 나타날 수도 있고 탭이 달린 형태로 다른 네비게이터가 함께 나타날 수도 있다. 탭이 달린 노트북의 일부인 네비게이터를 활성화하려면 해당 탭을 클릭한다. Workbench는 탭이 화면의 상단 또는 하단에 있는지 여부를 포함하여 편리하고 신속한 여러 환경 구성 방법을 제공한다.

다음은 AnyLink 네비게이터의 초기화면이다.



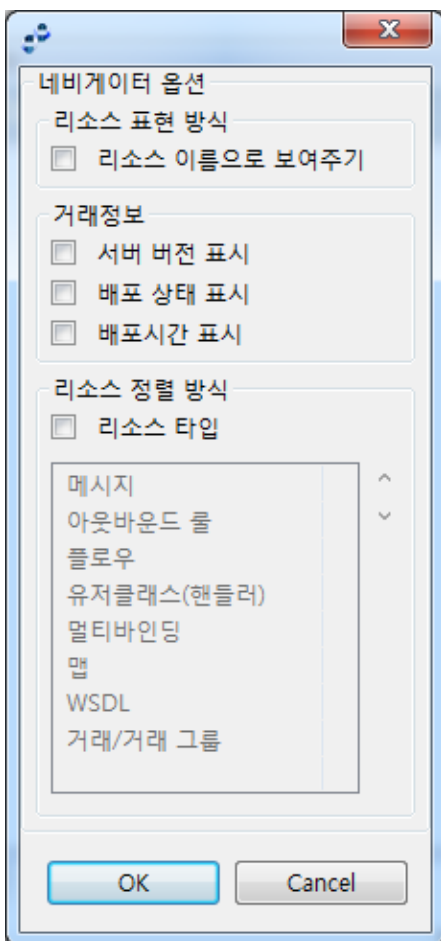
AnyLink 네비게이터

다음은 네비게이터 컨텍스트 메뉴에 대한 설명이다.

메뉴	설명
[새로 만들기]	AnyLink에서 제공하는 서비스(거래, 플로우, 아웃바운드 등)를 생성할 수 있는 화면을 제공한다.
[다운로드]	서버와 연결이 가능한 상태에서 사용할 수 있으며, DIS에서 리소스 다운로드한다.
[템플릿]	공통 설정을 가진 리소스들을 여러 개를 빠르게 생성할 수 있는 템플릿을 생성하거나 DIS에 업로드할 수 있다.
[업무시스템 할당]	현재 활성화된 프로젝트를 배포할 때 해당 프로젝트의 리소스를 RTE(RunTime Engine Server)에 반영하는 메뉴이다. 로컬 모드에서 실행하면 로그인 화면 이 실행된다.
[업무시스템 할당 해제]	RTE에 반영되었던 리소스를 해제시키는 메뉴이다. 로컬 모드에서 실행하면 로그인 화면 이 실행된다.
[Configuration 가져오기]	DIS에 등록되어 있는 설정 파일을 가져와 설정 정보들을 스튜디오에 등록한다. (예: 거래 타입)
[새로고침]	Service Explorer 에서 보여주는 Workspace의 내용을 실제 파일 시스템의 내용으로 새로고침한다. Workspace 내의 프로젝트는 실제로 파일 시스템과 동일한 내용을 보여준다. Workbench 외부적으로 변경, 추가 또는 삭제된 파일에 대해서는 Workbench는 알지 못한다. 이러한 경우에 이 새로고침 명령을 사용함으로써 Workbench와 실제의 파일 시스템과 동기를 맞춘다.

메뉴	설명
[붙여넣기]	복사한 프로젝트 또는 파일을 붙여넣기 한다.
[삭제하기]	내부의 프로젝트 또는 파일을 삭제하는 메뉴이다.
[가져오기]	외부의 프로젝트 또는 파일을 네비게이터 내부로 가져온다.
[내보내기]	내부의 프로젝트 또는 파일을 외부로 내보낸다.
[Validate]	소스 유효성 검사를 수행한다.
[Team]	SVN 등과 같은 시스템과 연동하여 팀 단위의 소스를 관리한다.
[Compare With]	리소스들을 비교한다.
[Replace With]	History를 이용하여 리소스를 변경한다.

 버튼을 클릭하면 리소스 표현 방식을 설정할 수 있는 다이얼로그가 생성된다.



네비게이터 옵션 - 리소스 표현 방식

다음은 네비게이터 리소스 표현 방식 변경방법에 대한 설명이다.

옵션	설명
리소스 이름으로 보여주기	옵션을 설정할 때 리소스 이름이 네비게이터에 표시된다. 옵션을 설정하지 않으면 리소스 아이디가 네비게이터에 표시된다.
서버 버전 표시	로그인 상태에서 거래(그룹) 리소스에 표시되는 정보로 DIS에 배포되어 있는 리소스의 버전을 출력한다.

옵션	설명
배포 상태 표시	로그인 상태에서 거래(그룹) 리소스에 표시되는 정보로 해당 리소스의 배포/배포해제 상태를 출력한다.
배포 시간 표시	로그인 상태에서 거래(그룹) 리소스에 표시되는 정보로 해당 리소스가 DIS에 마지막으로 배포된 시간을 출력한다.
리소스 정렬 방식	'리소스 타입' 항목을 선택한 경우 하단에 설정된 리소스 타입 순서로 정렬되며, '리소스 타입' 항목을 선택하지 않은 경우 리소스 종류에 관계 없이 리소스 이름 순으로 정렬이 된다.

1.3. 리소스

리소스를 생성하기 위해 활성화된 왼쪽 네비게이터의 컨텍스트 메뉴에서 **[새로 만들기]**를 선택한다. 리소스 생성에 대한 세부 사항은 각 절에서 설명한다.

다음은 AnyLink에서 생성할 수 있는 리소스에 대한 설명이다.

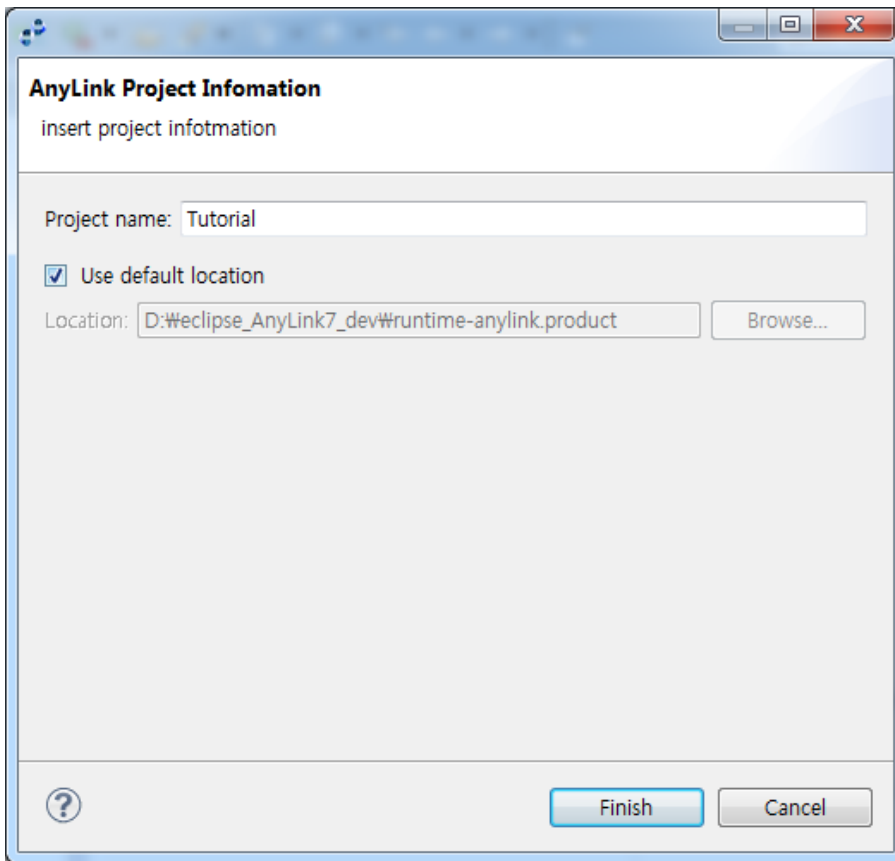
리소스	설명
프로젝트	<p>서비스 플로우 다이어그램을 그리려면 먼저 Workspace에 AnyLink 프로젝트를 생성해야 한다.</p> <p>AnyLink 프로젝트는 AnyLink 리소스를 관리하기 위한 최상위 단위이다. AnyLink 프로젝트 하위의 리소스들은 상호참조 및 연관을 맺을 수 있으며, 패키지 단위로 통일된다.</p>
거래/거래그룹	<p>거래는 AnyLink에서 제공하는 비즈니스 서비스이다.</p> <p>클라이언트로부터 요청 메시지를 받아서 백엔드 서비스를 호출하고 결과를 받아 응답 메시지를 전달하기까지 일련의 처리를 의미한다. 거래/거래그룹 리소스를 이용하여 거래그룹, 트리, 메시지 등을 생성/수정할 수 있다.</p>
멀티 바인딩	<p>멀티바인딩 라우터 룰은 AnyLink의 서비스 중 하나로 어댑터의 파싱 룰을 통해서 멀티바인딩 룰을 호출할 수 있다.</p> <p>AnyLink 내부 서비스를 경우에 따라 다르게 라우팅을 해야 할 경우 여러 서비스를 묶어서 하나의 그룹으로 조회한다.</p>
아웃바운드 룰	<p>AnyLink의 내부 서비스로 AnyLink에서 외부 시스템을 호출하고, 그 결과 응답을 받는 역할을 한다.</p> <p>내부적으로 다른 컴포넌트에서 아웃바운드 룰 서비스를 호출하면 해당 룰에 설정된 어댑터와 엔드포인트를 식별하여 외부 시스템으로 메시지를 전송하는 방식으로 동작한다.</p>

리소스	설명
WSDL	<p>WSDL(Web Services Description Language)는 메시지를 주고받으며 동작하는 엔드포인트들로 네트워크 서비스의 인터페이스를 기술하는 XML 포맷이다. 오퍼레이션과 메시지는 추상적으로 정의하고 네트워크 프로토콜과 메시지 포맷을 바인딩하여 실제 엔드포인트의 인터페이스를 기술한다.</p> <p>또한 WSDL은 특정 메시지 포맷과 네트워크 프로토콜에 제한되지 않고 확장하여 다양한 네트워크 인터페이스를 나타낼 수 있도록 구성되어 있다.</p>
외부 매핑	메시지 간 DataObject의 매핑을 정의하는 리소스이다.
플로우	<p>서비스 플로우는 비즈니스 프로세스를 모델링하는데 사용되는 표준인 BPMN(Business Process modeling Notation)을 차용한 다이어그램을 통해 프로세스의 흐름을 표현한다.</p> <p>AnyLink 스튜디오를 통해 액티비티와 이벤트로 구성된 다이어그램 형태로 정의하며, 프로세스 흐름 외에 변수, 표현식, 매핑 등의 기능을 제공한다. 즉, 서비스 플로우는 서비스의 수행 절차를 그래프의 형태로 표현한 것이다.</p>
유저클래스, 핸들러	거래에서 플로우는 제어의 흐름을 구조화하기 위하여 조건(condition)을 기술한다. 이런 조건은 표현식을 통해 하나의 문장형태로 프로세스 내의 변수를 이용하여 정의한다. 표현식으로 정의할 수 있는 로직보다 복잡한 로직을 정의해야 할 경우 유저 클래스 액티비티 또는 핸들러를 정의한다.
메시지	메시지는 서비스 간의 데이터를 교환하는데 사용하는 기본 데이터 형태로 전문을 추상화(abstraction)시킨 객체이다.
공통 업무 프로젝트	동일한 업무시스템을 가지는 프로젝트에 대해서 공통으로 사용할 수 있는 리소스들의 집합이다.

1.4. 프로젝트 생성

AnyLink의 리소스를 생성하기 전에 AnyLink의 프로젝트를 먼저 생성해야 한다.

왼쪽 네비게이터의 컨텍스트 메뉴에서 **[새로 만들기] > [프로젝트]**를 선택하면 **프로젝트 생성 화면**에 나타난다. 각 항목을 입력하고 **[Finish]** 버튼을 클릭한다.



프로젝트 생성 화면

항목	설명
Project name	프로젝트 이름을 입력한다. 프로젝트 이름으로 특수문자(?, <, >, ", *,)는 쓸 수 없다.

2. 거래/거래그룹 에디터

본 장에서는 AnyLink 스튜디오에 내장된 거래/거래그룹 에디터의 기능과 사용법에 대해 설명한다.

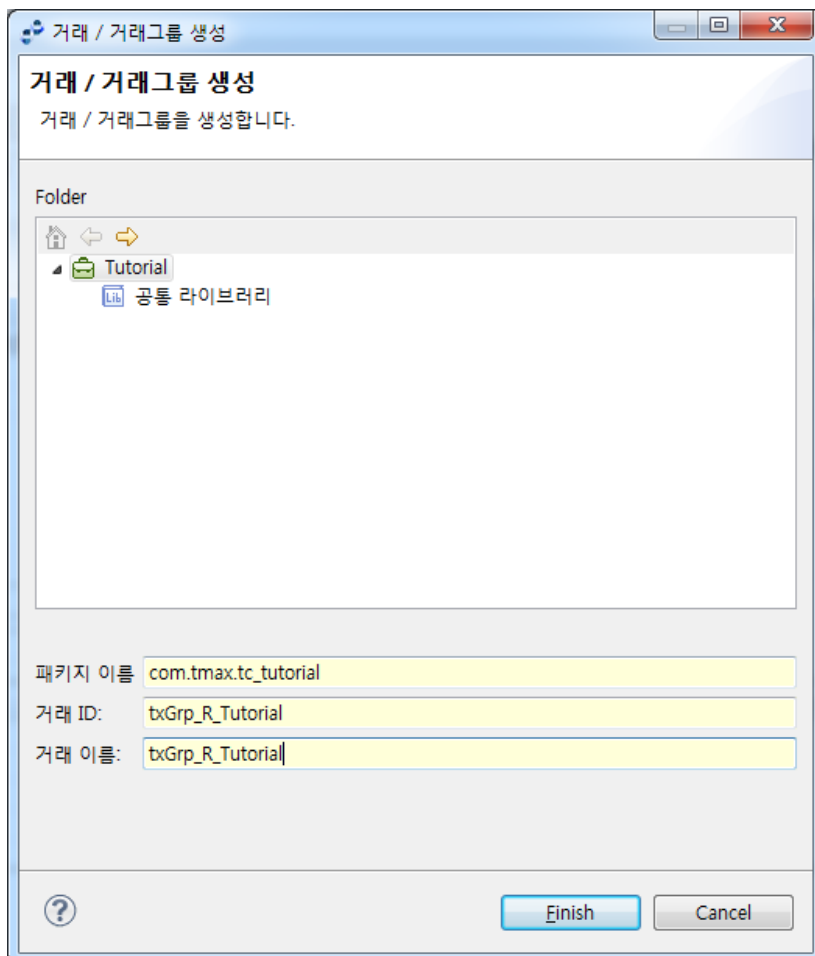
2.1. 거래/거래그룹 생성

거래/거래그룹 정보는 AnyLink의 RTE(RunTime Engine Server)의 실행에 필요한 비즈니스 리소스 중 하나로 업무 또는 거래와 관련되어 있다. 거래와 거래그룹으로 구성되어 있고 트리구조를 이룬다. 거래 및 거래구조는 요청 메시지, 응답 메시지, 비정상 응답 메시지로 이루어져 있고, 거래 트리의 기본 목적은 요청 메시지를 파싱하여 메시지 내용에 맞는 AnyLink의 엔진 내부 서비스를 호출하는 것이다.

거래 및 거래그룹은 노드 타입에 따라 최상위 거래그룹(Root), 거래그룹(Middle), 거래(Leaf)로 구분이 된다.

최상위 거래그룹 생성

최상위 거래그룹을 생성하기 위해 네비게이터의 **프로젝트** 리소스의 컨텍스트 메뉴에서 **[새로만들기] > [거래 / 거래그룹]**을 선택한다. **거래/거래그룹 생성 화면**의 각 항목을 입력하고 **[Finish]** 버튼을 클릭한다.

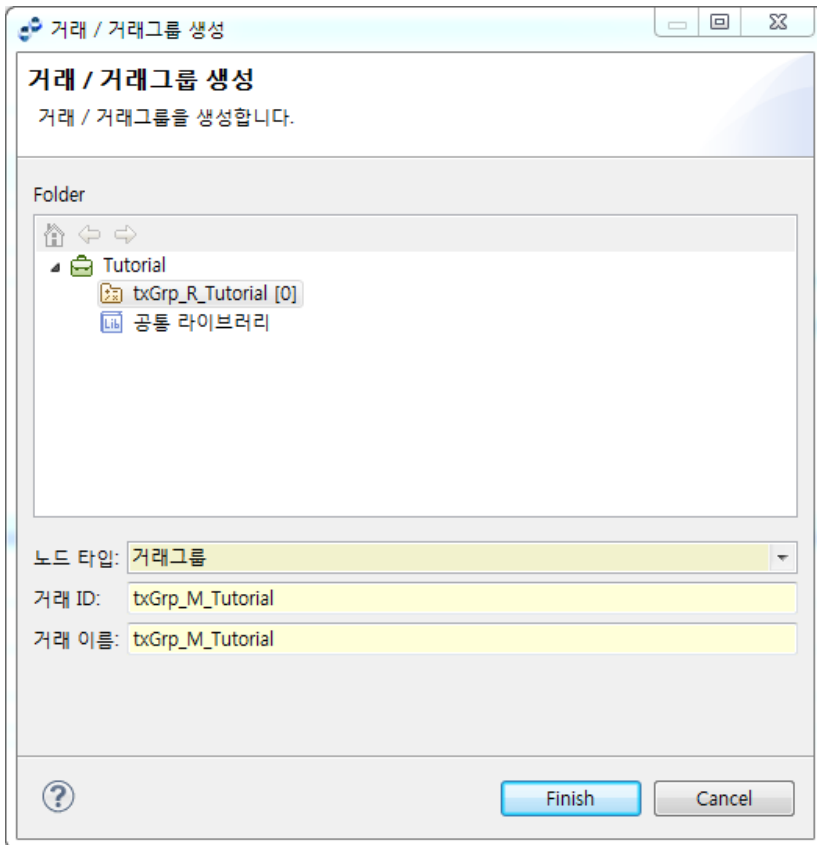


거래/거래그룹 생성 화면 - 최상위 거래그룹(Root)

항목	설명
패키지 이름	Java와 같이 해당 거래가 저장될 패키지 이름을 입력한다. 패키지명과 같이(예: com.tmax.tc_tutorial) 자동으로 디렉터리를 생성해 준다.
거래 ID	거래 리소스 아이디를 입력한다. 영어, 숫자, 특수문자(_)만 사용이 가능하며 첫 글자는 소문자로 입력한다.
거래 이름	거래 이름을 입력한다. 한글, 영어, 숫자, 특수문자 입력이 가능하다. 거래 이름은 XML Naming Convention을 따른다.

거래/거래그룹 생성

거래/거래그룹을 생성하기 위해 네비게이터의 **거래그룹** 리소스의 컨텍스트 메뉴에서 **[새로만들기] > [거래/거래그룹]**을 선택한다. **거래/거래그룹 생성 화면**의 각 항목을 입력하고 **[Finish]** 버튼을 클릭한다.



거래/거래그룹 생성 화면 - 거래그룹(Middle), 거래(Leaf)

항목	설명
노드 타입	노드 타입에 따라 '거래그룹', '거래'를 선택한다.
거래 ID	거래 리소스 아이디를 입력한다. 영어, 숫자, 특수문자(_)만 사용이 가능하며 첫 글자는 소문자로 입력한다.

항목	설명
거래 이름	거래 이름을 입력한다. 한글, 영어, 숫자, 특수문자 입력이 가능하다. 거래 이름은 XML Naming Convetion을 따른다.

2.2. 거래/거래그룹 에디터

거래/거래그룹 에디터는 **거래(그룹) 정보**, **거래(그룹)옵션**, **파싱 정보**, **파싱 옵션**, **XML 뷰어** 탭으로 구성된다. 각 항목에 들어가기 위해서는 리소스 View의 하단에 있는 각 항목 탭을 클릭한다.

2.2.1. 거래/거래그룹 정보

[거래/거래그룹 정보] 탭은 **기본 정보**, **요청 메시지**, **정상 응답 메시지**, **업무오류 응답 메시지**, **오류 응답 처리**, **호출 서비스**로 구성된다. 호출 서비스 거래에만 존재한다.

기본 정보

기본정보

- 거래그룹 ID
- 패키지 이름
- 거래그룹 이름
- 거래 타입

txGrp_R_Tutorial

com.tmax.tc_tutorial

txGrp_R_Tutorial

NONE

- 버전
- 설명
- Bypass 설정
- XA 설정

0

BizTx Group

NO

NO

[거래(그룹) 정보] - [기본 정보]

항목	설명
거래그룹 ID	거래그룹의 아이디로 거래그룹을 생성할 때 입력한 내용이 들어간다.
패키지 이름	거래그룹을 생성할 때 입력한 내용이 들어간다.
거래그룹 이름	거래그룹의 이름으로 거래그룹을 생성할 때 입력한 내용이 들어간다.
거래 타입	거래(그룹)의 타입을 설정한다. 모니터링할 때 거래 종류별 분류를 위해 설정하는 값이다. [Configuration 가져오기] 메뉴를 통해 거래 타입 등록이 가능하다.
버전	배포할 때 버전이 자동으로 변경된다.
설명	해당 거래그룹에 대한 설명을 입력한다. 사용자 편의를 위한 것으로 입력하지 않아도 무방하다.

항목	설명
ByPass 설정	<p>거래에 하나의 백엔드서비스(Backend Service) 호출이 있고 거래 요청 메시지와 서비스 호출 메시지가 동일한 경우로 메시지 변환없이 단순 메시지만 전달한다. 호출 서비스에 ByPass가 설정되어 있는 아웃바운드 룰만 설정이 가능하다.</p> <ul style="list-style-type: none"> ◦ PARENT : 상위 거래그룹의 설정과 동일하게 유지한다. ◦ YES : ByPass 설정을 활성화시킨다. ◦ NO : ByPass 설정을 비활성화시킨다.
XA 설정	<p>글로벌 트랜잭션(Global Trascation)을 지원하기 위한 설정이다.</p> <p>트랜잭션을 지원하는 어댑터(Tmax, WebService 등)를 통해 prepare, commit, rollback 등의 글로벌 트랜잭션 2-phase commit 관련 메시지를 처리하여 하나의 트랜잭션으로 묶여 있는 여러 건의 메시지를 일관 처리할 수 있도록 한다.</p> <ul style="list-style-type: none"> ◦ PARENT : 상위 거래그룹의 설정과 동일하게 유지한다. ◦ YES : XA 설정을 활성화시킨다. ◦ NO : XA 설정을 비활성화시킨다.

거래 호출 서비스(거래)

다음은 거래/거래그룹 중 거래인 경우에 **[거래정보]** 탭의 **거래 호출 서비스** 영역에 대한 설명이다. 설정한 서비스 타입에 따라 **플로우**, **아웃바운드 룰**, **멀티바인딩 서비스**를 호출한다.

• 플로우

호출 서비스

- 서비스 타입: FLOW
- 서비스 이름: SF_L_A_Tutorial (SF_L_A_Tutorial_Message_? 검색)
- 매핑 설정: ☒
- 요청 매핑: Tx_L_A_tutorialCallServiceInputMap.map 검색 생성
- 정상 응답 매핑: 검색 생성
- 업무오류 응답 매핑: 검색 생성

[거래(그룹) 정보] - [호출 서비스(플로우)]

항목	설명
서비스 타입	플로우를 호출하기 위해 'FLOW'를 선택한다.
서비스 이름	[검색] 버튼을 클릭해서 생성되는 다이얼로그에서 호출할 플로우를 선택한다.

항목	설명
매핑 설정	<p>거래의 요청/정상 응답/업무오류 응답 메시지와 플로우 Start Event의 Input/Output/Error 파라미터가 달라 메시지와 파라미터 사이에 매핑이 필요한 경우 선택한다.</p> <ul style="list-style-type: none"> 요청 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 거래/거래그룹의 요청 메시지와 플로우 Start Event의 Input 파라미터의 입력 매핑을 생성한다. 정상 응답 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 플로우 Start Event의 Output 파라미터와 거래/거래그룹의 정상 응답 메시지의 출력 매핑을 생성한다. 업무오류 응답 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 플로우 Start Event의 Error 파라미터와 거래/거래그룹의 업무오류 응답 메시지의 출력 매핑을 생성한다.

• 아웃바운드 룰

호출 서비스

- 서비스 타입: OUTBOUND RULE
- 서비스 이름: OutB_L_A_TCP_Tutorial 🔍 검색
- 비동기 응답 거래: ☒
- 코릴레이션: ☒

메시지 ID	필드 ID
Req_Header_R_Tutorial	branch_number

+ 추가
- 삭제

- 매핑 설정: ☐

[거래(그룹) 정보] - [호출 서비스(아웃바운드 룰)]

항목	설명
서비스 타입	<p>아웃바운드 룰을 호출하기 위해 'OUTBOUND RULE'을 선택한다.</p> <p>By Pass 설정된 거래의 경우 'OUTBOUND RULE'만 선택 가능하다.</p>
서비스 이름	<p>[검색] 버튼을 클릭하면 생성되는 다이얼로그에서 호출할 아웃바운드 룰을 선택한다. By Pass 설정된 거래의 경우 By Pass가 설정된 아웃바운드 룰만 선택 가능하다.</p>

항목	설명
비동기 응답 거래	<p>비동기 응답 거래를 사용하기 위해서는 [파싱 정보] > [파싱 정보 정의]에 '거래 식별코드'가 'RESPONSE' 또는 'ERROR_RESPONSE'로 설정되어 있어야 한다.</p> <p>다음은 '거래 식별코드'에 따른 설명이다.</p> <ul style="list-style-type: none"> 'RESPONSE' 혹은 'ERROR_RESPONSE'로 설정된 거래의 경우 : 해당 거래에 대한 응답 유무와 상관 없이 서비스를 수행할 수 있다. 단, 서비스 타입이 응답 수신 방식이 ASYNC인 TCP 아웃바운드 룰로 설정되어 있어야 하며, 다른 응답 수신 방식을 가지는 경우 비동기 응답 거래를 선택할 때 ASYNC 타입으로 변경된다. 'RESPONSE'로 설정된 경우 : TCP 아웃바운드 룰의 정상 응답 메시지에 거래의 요청 메시지가 추가된다. 'ERROR_RESPONSE'로 설정된 경우 : TCP 아웃바운드 룰의 업무오류 응답 메시지에 거래의 요청 메시지가 추가된다. <p>비동기 응답 거래를 사용하는 경우는 매핑 설정 옵션 사용이 불가능하다.</p>
코릴레이션	<p>비동기 응답 거래로 설정된 경우 사용 가능하며 '거래 식별코드'의 종류에 따라 'RESPONSE'의 경우 정상 응답 메시지, 'ERROR_RESPONSE'의 경우 업무오류 응답 메시지의 코릴레이션으로 사용할 메시지 필드를 선택한다. 선택된 메시지 필드는 룰의 요청 메시지 코릴레이션 필드와 매칭되어 사용된다.</p> <p>By Pass 설정된 거래의 경우 코릴레이션은 설정하지 않는다.</p>
매핑 설정	<p>거래의 요청/정상 응답/업무오류 응답 메시지와 아웃바운드 룰의 요청/응답/업무오류 응답 메시지 사이에 매핑이 필요한 경우 선택한다.</p> <ul style="list-style-type: none"> 요청 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 거래/거래그룹의 요청 메시지와 아웃바운드 룰의 요청 메시지의 입력 매핑을 생성한다. 정상 응답 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 아웃바운드 룰의 응답 메시지와 거래/거래그룹의 정상 응답 메시지의 출력 매핑을 생성한다. 업무오류 응답 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 아웃바운드 룰의 업무오류 응답 메시지와 거래/거래그룹의 업무오류 응답 메시지의 출력 매핑을 생성한다. <p>By Pass 설정된 거래의 경우 매핑 설정은 하지 않는다.</p>

• 멀티바인딩

호출 서비스

서비스 타입

MULTIBINDING

서비스 이름

MBinding_L_A_Tutorial

Q 검색

매핑 설정

☒

요청 매핑

Tx_L_A_tutorialCallServiceInputMap.map

Q 검색

📄 생성

정상 응답 매핑

Q 검색

📄 생성

업무오류 응답 매핑

Q 검색

📄 생성

[거래(그룹) 정보] - [호출 서비스(멀티바인딩)]

항목	설명
서비스 타입	멀티바인딩을 호출하기 위해 'MULTIBINDING'을 선택한다.
서비스 이름	[검색] 버튼을 클릭해서 생성되는 다이얼로그에서 호출할 멀티바인딩을 선택한다.
매핑 설정	<p>거래의 요청/정상 응답/업무오류 응답 메시지와 멀티바인딩의 요청/응답/업무오류 응답 메시지 사이에 매핑이 필요한 경우 선택한다.</p> <ul style="list-style-type: none"> 요청 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 거래/거래그룹의 요청 메시지와 멀티바인딩의 요청 메시지의 입력 매핑을 생성한다. 정상 응답 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 멀티바인딩의 응답 메시지와 거래/거래그룹의 정상 응답 메시지의 출력 매핑을 생성한다. 업무오류 응답 매핑 : [검색] 버튼을 클릭해서 미리 생성된 매핑 파일을 선택하거나 [생성] 버튼을 클릭해서 멀티바인딩의 업무오류 응답 메시지와 거래/거래그룹의 업무오류 응답 메시지의 출력 매핑을 생성한다.

요청 / 정상응답 / 업무오류 응답 메시지

요청 / 정상응답 / 업무오류 응답 메시지는 거래/거래그룹을 구성하는 메시지이다. 거래그룹에 메시지를 추가할 경우 하위 노드의 메시지 정의에 상속되어 공통적으로 적용된다.

요청 메시지

이름	메시지 아이디	타입 아이디
Req_Header_R_...	Req_Header_R_Tuto...	Req_Header_R_Tut...

+ 추가
 - 삭제

- 네임스페이스URI
- 로컬 파트
- 그룹
- 그룹번호

항목	설명
요청 메시지	거래에 요청 메시지를 설정한다.
정상 응답 메시지	거래에 정상 응답 메시지를 설정한다.
업무오류 응답 메시지	거래에 업무오류 응답 메시지를 설정한다. 시스템적으로 발생한 오류가 아닌 업무적으로 발생한 오류를 의미한다. 시스템적으로 발생한 오류는 "오류 응답처리"에서 처리한다.

다음은 요청 메시지를 추가하는 다이얼로그 화면이다.

왼쪽 네비게이터에서 **프로젝트, 거래(그룹)**을 선택하면 **프로젝트, 거래(그룹)**에 포함된 메시지가 오른쪽 테이블에 나타난다. 메시지를 선택하면 해당 메시지가 오른쪽 테이블에 나타난다. 오른쪽 테이블에서 메시지 타입 변경이 가능하며 메시지를 선택하면 해당 메시지가 요청 메시지에 등록된다.



AnyLink 리소스는 거래 노드 단위로 계층 구조를 가지고 있기 때문에 상위 거래 노드에 포함된 리소스는 하위 거래 노드의 리소스를 참조할 수 없다.

업무시스템 검색을 선택할 때 네비게이터 내에 존재하는 동일한 업무시스템을 가진 프로젝트의 메시지 검색이 가능하다. Search에 검색할 단어를 입력하면 자동으로 검색되며, 입력한 단어를 포함하는 메시지 ID와 메시지 이름을 가진 메시지들이 검색 된다.

업무시스템 검색을 통해 다른 프로젝트의 메시지를 선택하는 경우 해당 메시지가 포함되어 있는 거래 노드가 Symbolic Link로 생성된다. 거래를 배포하기 위해서는 Symbolic Link로 생성된 거래 노드가 먼저 배포되어 있어야 한다.

다음은 **요청 메시지, 정상 응답 메시지, 업무오류 응답 메시지**에서 공통으로 정의해야 하는 항목들이다.

항목	설명
네임스페이스 URI	요청 / 정상 응답 / 업무오류 응답 메시지로 설정된 메시지가 XML 타입인 경우 사용되며 로컬 파트와 함께 설정된 메시지의 네임스페이스를 구성한다.
로컬 파트	요청 / 정상 응답 / 업무오류 응답 메시지로 설정된 메시지가 XML 타입인 경우 사용되며 네임스페이스 URI와 함께 설정된 메시지의 네임스페이스를 구성한다.
그룹	SOAP 헤더의 거래에서 사용되며 XML에서 태그를 생성한다.
그룹 번호	그룹의 순서를 설정한다.

오류 응답 처리

오류 응답 처리 방식을 설정할 수 있다.

- **NONE**

아무 처리도 하지 않는다.

- **PARENT**

상위 거래 노드의 처리 방식과 동일하게 처리한다.

- **FLOW**

오류 응답 방식을 FLOW로 설정할 경우 다음의 탭이 생성된다. 각 탭들은 오류 응답의 종류를 나타낸다.

탭	설명
[시스템 오류응답]	시스템적으로 발생한 오류에 대한 응답을 처리한다.

탭	설명
[포맷 오류응답]	메시지를 DataObject로 변환할 때 Data Type이 맞지 않아 발생하는 오류에 대한 응답을 처리한다.
[거래제한 응답]	WebAdmin 거래 제어 설정에 따른 거래 제한에 대한 응답을 처리한다.
[파싱 오류응답]	하위 노드를 찾지 못한 경우에 발생하는 오류에 대한 응답을 처리한다.
[중복거래 오류응답]	중복거래 확인 여부 옵션이 설정되어 있고, 중복거래 확인 필드에 설정한 값이 동일한 거래가 여러 번 수행되는 경우 첫 번째 거래를 제외한 거래들을 중복 거래로 판단하여 오류처리한다.
[서비스 오류응답]	플로우 혹은 아웃바운드 룰이 Throw한 오류에 대한 응답을 처리한다.
[코릴레이션 오류응답]	플로우 코릴레이션 오류가 발생하는 경우 오류에 대한 응답을 처리한다.

오류 응답 종류에 맞는 '오류 응답 메시지'를 체크할 경우 해당 오류 응답 메시지를 설정할 수 있다. FLOW를 선택했으므로 설정한 오류가 발생했을 때 '서비스 ID'로 등록된 서비스 플로우가 호출된다.

오류 응답처리

• 오류 응답 방식 FLOW

시스템 오류응답 | 포맷 오류응답 | 거래제한 응답 | 파싱 오류응답 | 중복거래 오류응답 | 서비스 오류응답 | 코릴레이션 오류응답

☒ 오류 응답 메시지

이름	ID	타입
Req_Header_R_Tutorial	Req_Header_R_Tutorial	Req_Header_R_TutorialFi

서비스 ID: 검색

입력매핑 ID: 검색 생성

출력 매핑 ID: 검색 생성

네임스페이스 URI:

로컬 파트:

+ 추가 - 삭제

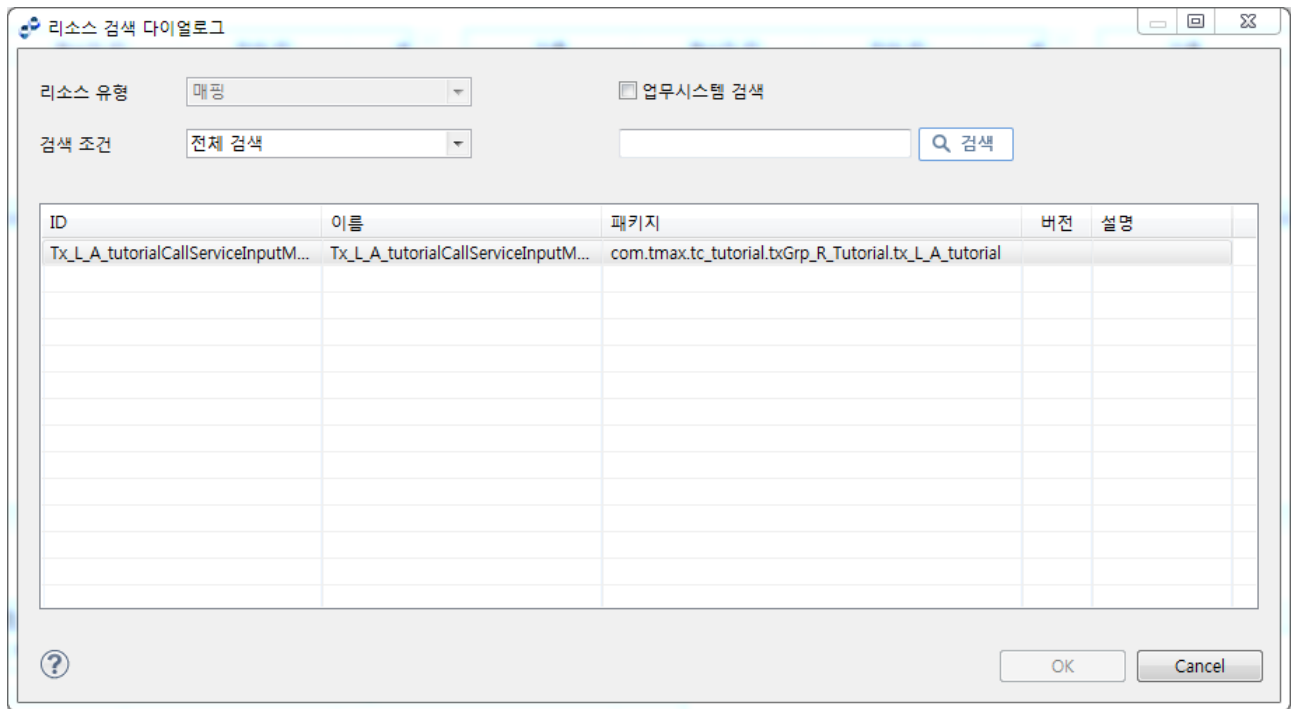
[거래(그룹) 정보] - [오류 응답 방식(FLOW)]

다음은 각 항목에 대한 설명이다.

항목	설명
오류 응답 메시지	<ul style="list-style-type: none"> 오류 응답 메시지 : 오류 응답을 설정하기 위한 체크박스이다. 체크하지 않은 경우 오류 응답을 설정할 수 없다. [추가] 버튼 : [거래(그룹) 정보] - [요청 메시지]에서 서비스 플로우의 출력 메시지와 매핑시킬 메시지를 선택할 수 있다. [제거] 버튼 : 추가한 메시지를 목록에서 제거한다. 네임스페이스 URI : 설정된 메시지가 XML 타입인 경우 사용되며 로컬 파트와 함께 설정된 메시지의 네임스페이스를 구성한다. 로컬 파트 : 설정된 메시지가 XML 타입인 경우 사용되며 네임스페이스 URI와 함께 설정된 메시지의 네임스페이스를 구성한다.
서비스 ID	[검색] 버튼을 클릭해서 오류가 발생하는 경우 호출할 서비스 플로우를 선택한다. 해당 거래에 각 탭에 맞는 오류가 발생하게 되면 설정한 서비스 플로우가 호출된다.

항목	설명
입력매핑 ID	<p>'서비스 ID'로 등록한 서비스 플로우에 들어갈 메시지를 검색/생성한다.</p> <ul style="list-style-type: none"> • [검색] 버튼 : Map 파일 리스트를 보여주는 검색 다이얼로그(매핑 아이디 검색 다이얼로그(리소스 검색))에서 메시지를 선택한다. • [생성] 버튼 : 매핑한 결과가 Map 파일로 생성되며 네비게이터의 거래/거래그룹 리소스의 컨텍스트 메뉴 [새로만들기] > [외부 매핑]을 선택해도 생성이 가능하다. <p>[생성] 버튼을 클릭하면 매핑 다이얼로그가 생성되며 요청 메시지에 등록한 메시지와 '서비스 ID'에서 선택한 플로우 Start 메시지 이벤트의 Input 파라미터로 설정된 메시지를 매핑한다.</p> <p>'Source'에는 요청 메시지에 등록한 메시지를 컨텍스트 메뉴의 [Add Source] 버튼을 클릭해서 등록 가능하며 'Target'에는 플로우 Start 메시지 이벤트의 Input 파라미터로 설정된 메시지가 등록되어 있다.</p> <p>시스템, 포맷, 파싱 오류 응답의 경우 Source로 설정될 수 있는 해당 거래 노드의 DataObject가 생성되지 않아 [Add Source] 버튼 클릭할 때 상위 노드의 메시지만 보여진다.</p> <p>[Add Mapping Code] 버튼을 클릭하여 Mapping Code를 직접 작성하여 사용도 가능하다. '서비스 ID'가 설정되지 않은 경우 매핑 다이얼로그가 생성되지 않는다.</p>
출력매핑 ID	<p>'서비스 ID'로 등록한 서비스 플로우를 마치고 출력 메시지로 나오는 메시지를 검색/생성한다.</p> <ul style="list-style-type: none"> • [검색] 버튼 : Map 파일 리스트를 보여주는 검색 다이얼로그(매핑 아이디 검색 다이얼로그(리소스 검색))에서 메시지를 선택한다. • [생성] 버튼 : 매핑 다이얼로그가 생성되며 '서비스 ID'에서 선택한 플로우 Start 메시지 이벤트의 Input 파라미터로 설정된 메시지와 오류 응답 메시지에 등록한 메시지를 매핑한다. <p>Source에는 '서비스 ID'에서 선택한 플로우 Start 메시지 이벤트의 Input 파라미터로 설정된 메시지를 컨텍스트 메뉴의 [Add Source] 버튼을 클릭해서 등록 가능하며 Target에는 오류 응답 메시지에 등록한 메시지가 등록되어 있다. '서비스 ID'가 설정되지 않은 경우 매핑 다이얼로그가 생성되지 않는다.</p>

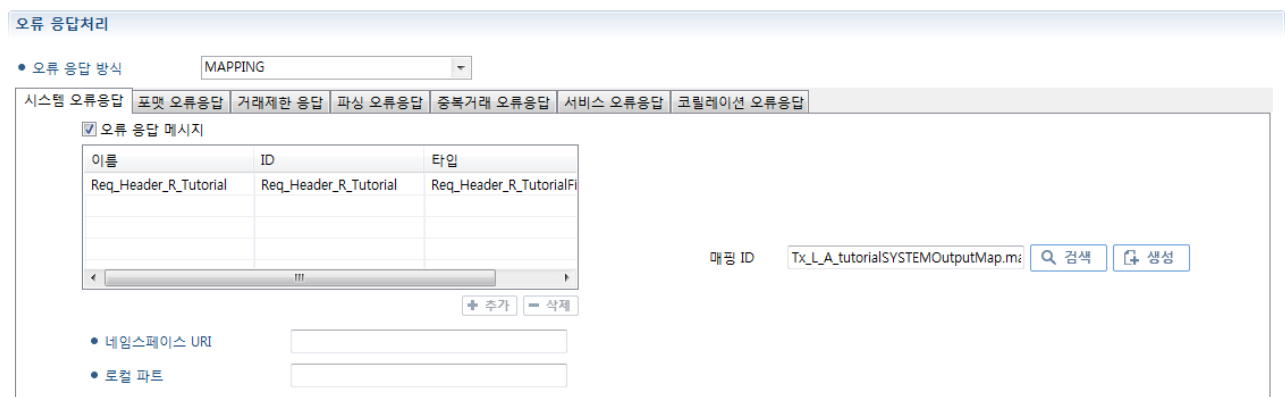
등록할 Map 파일 목록을 조회한 후 **[OK]** 버튼을 클릭한다.



매핑 아이디 검색 다이얼로그(리소스 검색)

• MAPPING

오류 응답에 관한 메시지를 MAPPING으로 넘긴다. 각 오류 응답에 해당하는 전문을 출력 매핑으로 정의한다.



[거래(그룹) 정보] - [오류 응답 방식(MAPPING)]

항목	설명
오류 응답 메시지	"Flow 오류 응답 방식"과 동일하다.
매핑 ID	<ul style="list-style-type: none"> [검색] 버튼 : Map 파일 리스트를 보여주는 검색 다이얼로그(매핑 아이디 검색 다이얼로그(리소스 검색))에서 메시지를 선택한다. [생성] 버튼 : 매핑 다이얼로그가 생성되며 요청 메시지에 등록된 메시지와 오류 응답 메시지에 등록된 메시지를 매핑한다. 'Source'에는 요청 메시지에 등록된 메시지를 컨텍스트 메뉴의 [Add Source] 버튼을 클릭해서 등록 가능하며 'Target'에는 오류 응답 메시지에 등록된 메시지가 등록되어 있다. 시스템, 포맷, 파싱 오류 응답의 경우 Source로 설정될 수 있는 해당 거래 노드의 DataObject가 생성되지 않아 [Add Source] 버튼 클릭할 때 상위 노드의 메시지만 보여진다. [Add Mapping Code] 버튼을 클릭하여 Mapping Code를 직접 작성하여 사용도 가능하다.

2.2.2. 거래/거래그룹 옵션

[거래(그룹) 옵션] 탭에서 거래/거래그룹에 대한 타임아웃, 연결 고정, GUID, 응답 유무, 전달 보장, 중복거래 거래 우선순위 관련 옵션을 설정할 수 있다.

거래 옵션 정의

- 거래 플로우 타임아웃(ms)
- 타임아웃 설정 안함 ☒
- 연결 고정
- 외부 GUID 사용
- GUID 타입
- 응답 유무
- 전달 보장 설정 ☐
 - 전달 보장
 - 큐 ID
 - 개별 설정 ☐
 - 재시도 간격(s)
 - 최대 재시도 횟수
 - 만료 시간(s)

- 중복거래 확인 여부
 - 중복거래 확인 필드

메시지 ID	필드 ID
- 중복거래 보관 주기
- 거래 우선순위

메시지 ID 필드 ID

값	우선순위

[거래(그룹) 옵션] - [거래(그룹) 옵션 정의]

항목	설명
거래 플로우 타임 아웃(ms)	<p>거래 플로우 타임아웃을 설정한다. 설정한 값이 지날 때까지 응답이 없으면 에러 처리된다.</p> <ul style="list-style-type: none"> 최상위 거래그룹이 아닌 경우 : '타임 아웃 설정 안함'을 선택하는 경우 상위 거래그룹의 타임 아웃 설정이 적용된다. 최상위 거래그룹의 경우 : 기본값이 1분(60000ms)으로 설정되고, 하위 거래 /거래그룹의 경우 '타임 아웃 설정 안함'이 선택되어 상위 노드의 설정과 동일하게 설정된다.
연결 고정	<p>플로우 코릴레이션 오류가 발생하는 경우 메시지를 다른 MS로 이동시킬지 여부를 설정하는 옵션이다.</p> <ul style="list-style-type: none"> PARENT : 상위 거래그룹의 설정을 따른다. YES : 코릴레이션 메시지를 다른 MS로 이동하지 않고 오류응답이 설정된 경우 오류응답 처리 후 세션을 종료한다. NO : 코릴레이션 메시지를 클러스터 된 다른 MS로 전달한다.

항목	설명
외부 GUID 사용	<p>외부로부터 받은 GUID를 AnyLink의 GUID로 사용할지 여부를 설정한다.</p> <ul style="list-style-type: none"> ◦ PARENT : 상위 거래그룹의 설정을 따른다. ◦ YES : 메시지 필드에 GUID 항목이 존재하는 경우 해당 필드값을 GUID로 사용한다. ◦ NO : AnyLink에서 자체 GUID를 생성하여 사용한다.
GUID 타입	<p>GUID 길이를 설정한다.</p> <ul style="list-style-type: none"> ◦ PARENT : 상위 거래그룹의 설정을 따른다. ◦ LONG : 19자리 이내의 long 데이터 값을 GUID로 생성한다. ◦ STRING : 32글자의 String 데이터 값을 GUID로 생성한다.
응답 유무	<ul style="list-style-type: none"> ◦ REQUEST_RESPONSE : 요청, 응답 메시지 외 응답에 대한 ACK 메시지가 추가된 형태이다. 응답 메시지를 잘 처리하였는지를 알려주는 ACK 또는 NAK를 전달해준다. ◦ PARENT : 상위 거래그룹의 설정을 따른다. ◦ ONEWAY : 응답을 기다리지 않고 요청 메시지만 전달한다. <p>응답 유무는 ByPass 거래인 경우에만 동작하는 옵션이다. ByPass 거래가 아닌 경우는 설정이 무시된다.</p>
전달 보장	<p>전달 보장을 사용할지에 대한 설정으로, 응답 유무가 ONEWAY인 경우에만 사용이 가능하다.</p> <ul style="list-style-type: none"> ◦ PARENT : 상위 거래그룹의 설정을 따른다. ◦ YES : 전달 보장을 사용한다. ◦ NO : 전달 보장을 사용하지 않는다.
큐 ID	<p>전달 보장에 사용할 큐를 설정한다.</p> <p>사용할 큐는 WebAdmin의 [구성관리] > [메시지 큐]에서 미리 생성해야 한다. [검색] 버튼을 클릭하면 설정할 큐 ID를 검색할 수 있는 다이얼로그가 나타난다.</p>
개별 설정	<p>큐 설정을 따르지 않고 거래의 큐 설정을 설정할 수 있다.</p> <ul style="list-style-type: none"> ◦ 재시도 간격(s) : 재시도 하는 시간 간격을 설정한다. ◦ 최대 재시도 횟수 : 메시지를 재시도하는 최대 횟수를 입력한다. -1을 입력할 경우 무한 재시도를 수행한다. ◦ 만료 시간(s) : 메시지가 만료되어 삭제되는 시간을 입력한다. -1을 입력할 경우 메시지가 만료되지 않는다.
중복거래 확인 여부	<ul style="list-style-type: none"> ◦ PARENT : 상위 거래그룹의 설정을 따른다. ◦ YES : 중복된 거래를 허용하지 않는다. 중복된 거래들은 중복거래 오류응답 루틴에 따라 처리된다. ◦ NO : 중복된 거래를 허용한다.

항목	설명
중복거래 확인 필드	<ul style="list-style-type: none"> ◦ 메시지 ID : 중복여부를 확인할 메시지의 아이디이다. ◦ 필드 ID : 중복여부를 확인할 필드의 아이디이다. <p>[+추가] 버튼을 클릭한 후 메시지와 필드를 선택하여 추가할 수 있다.</p> <p>테이블에서 삭제할 항목을 선택 후 [-삭제] 버튼을 클릭하여 삭제할 수 있다.</p>
중복거래 보관 주기	<p>중복확인을 위해 추출된 필드들을 보관하는 주기를 설정한다.</p> <ul style="list-style-type: none"> ◦ Daily : 1일마다 추출된 값들을 삭제한다. ◦ Hourly : 1시간 추출된 값들을 삭제한다.
거래 우선순위	<ul style="list-style-type: none"> ◦ PARENT : 거래그룹에서 ROOT가 아닐 경우 자신의 부모의 거래 우선순위를 따른다. ◦ HIGH : 거래그룹 중 해당 거래의 우선순위를 가장 높게 설정한다. WebAdmin에서 설정한 VIP ThreadPool에 할당된다. ◦ MEDIUM : 대부분의 거래가 MEDIUM의 우선순위를 가지며, 거래그룹 중 보통의 우선순위를 갖게 설정한다. VIP가 아닌 기본 ThreadPool에 할당된다. ◦ LOW : 크게 중요하지 않은 거래일 경우 LOW 우선순위를 설정한다. ThreadPool에 할당되나 Pool 넘칠 경우 삭제될 수 있다. ◦ MESSAGE : 설정한 메시지의 값에 따라 우선순위를 정할 수 있다. 우선순위는 MESSAGE를 제외한 기존의 우선순위와 같다.

'거래 우선순위'를 'MESSAGE'로 선택하면 다음과 같은 설정 화면이 나타난다.

• 거래 우선순위 MESSAGE

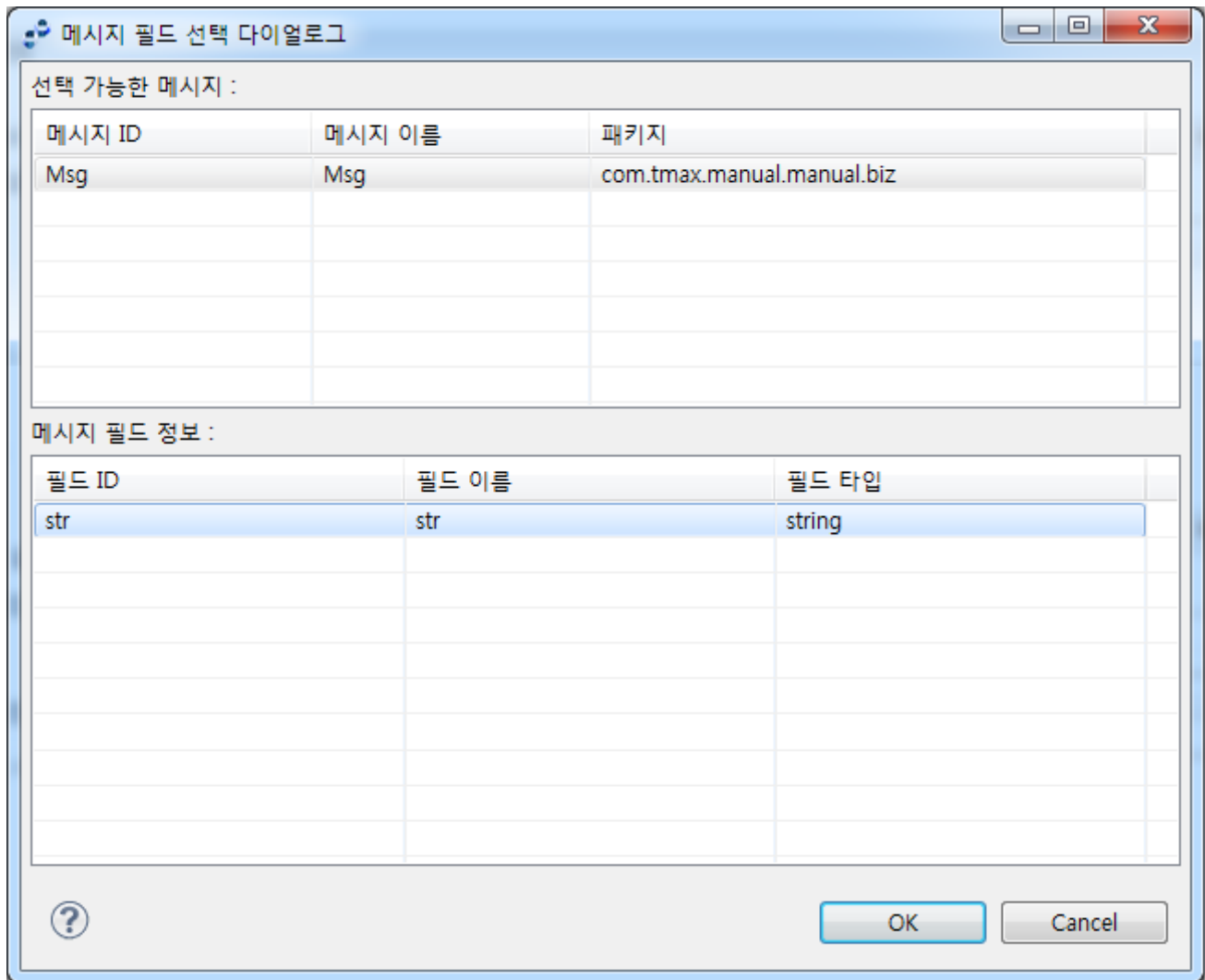
메시지 ID ReqBody 필드 ID data 🔍 검색

값	우선순위
value	MEDIUM

+ 추가 - 삭제

[거래(그룹) 옵션] - [거래 우선순위] - 메시지 설정

[검색] 버튼을 클릭하면 설정할 메시지를 검색할 수 있는 다이얼로그가 나타난다.



메시지 필드 선택 다이얼로그

2.2.3. 파싱정보

입력 전문이 처리되어야 할 하위 거래를 식별하기 위한 기준을 정의하는 기능을 한다.

• 거래 식별코드

거래 정보의 '**비동기 응답 거래**' 항목에 체크할 때 필수적으로 입력해야 한다.

거래 식별 코드

코드값	종류
0011	REQUEST

+ 추가 - 삭제

[파싱 정보] - [파싱 정보 정의] - 거래 식별코드

항목	설명
코드값	거래에 대한 식별코드를 입력한다.
종류	<ul style="list-style-type: none"> • RESPONSE : 정상 응답 처리 • REQUEST : 요청 처리 • ERROR_RESPONSE : 오류 응답 처리 <p>호출 서비스의 '비동기 응답 거래' 설정을 사용하기 위해서는 'RESPONSE' 또는 'ERROR_RESPONSE'의 설정이 필요하다.</p>

• 하위 거래 식별 방법

하위 거래 식별 방법은 거래그룹에서만 설정이 가능하다. 하위 거래를 식별하는 방법을 선택한다. [+추가] 버튼을 클릭한 후 값과 거래를 입력한다.

• 하위 거래 식별 방법 MESSAGE ▼

- 메시지 필드 선택

메시지 ID	필드 ID

+ 추가 - 삭제

- 거래 식별코드 매핑 ☒

값	거래 식별코드

+ 추가 - 삭제

[파싱 정보] - [파싱 정보 정의] - 하위 거래 식별 방법

항목	설명
NONE	하위 거래 식별을 하지 않는다.
MESSAGE	<p>선택한 메시지 필드값에 해당하는 거래 식별코드를 갖는 거래를 선택한다.</p> <p>메시지의 필드값이 '값'과 일치하는 경우 입력한 '거래 식별코드'를 갖는 거래를 선택한다.</p>
HANDLER	<p>주로 파싱 핸들러를 구현하는 경우 사용한다.</p> <p>파싱 핸들러에 해당하는 거래 식별코드를 갖는 거래를 선택한다.</p> <p>유저 클래스 반환값이 '값'과 일치하는 경우 입력한 '거래 식별코드'를 갖는 거래를 선택한다.</p>

항목	설명
X_PATH	XML Message를 사용할때 이 항목을 선택할 수 있다. X_PATH 값에 해당하는 거래 식별코드를 갖는 거래를 선택한다. X_PATH값이 '값'과 일치하는 경우 입력한 '거래 식별코드'를 갖는 거래를 선택한다.
OFFSET_LENGTH	해당 옵션을 선택하게 되면 Offset과 Length 입력창이 추가된다. OFFSET_LENGTH에 해당하는 거래 식별코드를 갖는 거래를 선택한다. OFFSET_LENGTH 값이 '값'과 일치하는 경우 입력한 '거래 식별코드'를 갖는 거래를 선택한다.
SOAP_ACTION	SOAP_ACTION에 해당하는 거래 식별코드를 갖는 거래를 선택한다. SOAP_ACTION이 '값'과 일치하는 경우 입력한 '거래 식별코드'를 갖는 거래를 선택한다.
JSON_POINTER	JSON POINTER에 해당하는 거래 식별코드를 갖는 거래를 선택한다. JSON POINTER이 '값'과 일치하는 경우 입력한 '거래 식별코드'를 갖는 거래를 선택한다.
CONDITION_MESSAGE	조건에 맞을 경우 하위 MESSAGE 설정에 따라 MESSAGE와 동일하게 동작하고, 조건이 맞지 않을 경우 다음 조건 탭을 확인한다. 조건은 선택한 메시지의 필드가 입력한 값이거나 조건이 없을 경우 true가 된다.
CONDITION_OFFSET_LENGTH	조건에 맞을 경우 하위 OFFSET_LENGTH 설정에 따라 OFFSET_LENGTH와 동일하게 동작하고, 조건이 맞지 않을 경우 다음 조건 탭을 확인한다. 조건은 선택한 메시지의 필드가 입력한 값이거나 조건이 없을 경우 true가 된다.
HTTP_METHOD	요청된 HTTP 메소드에 해당하는 거래 식별코드를 갖는 거래를 선택한다. HTTP 인바운드 엔드포인트로 요청된 거래에 대해서만 사용 가능하다.

2.2.4. 파싱 옵션

파싱 옵션에서 인코딩 타입과 메시지 타입을 설정한다.

- 인코딩 타입 설정

인코딩 타입 설정

☒ 인코딩 타입 사용

[+ 추가](#)

• 파싱옵션 선택

OFFSET_LENGTH

• Offset

0

• Length

10

[- 삭제](#)

옵션 값	인코딩
Option	UTF-8

[+ 추가](#)
[- 삭제](#)

인코딩 타입 설정 화면

항목	설명
인코딩 타입 사용	체크할 경우 파싱 옵션을 추가/삭제할 수 있다.
파싱옵션 선택	<ul style="list-style-type: none"> All: 해당 거래에 해당하는 모든 메시지에 대해 설정한 '인코딩 값'을 할당한다. ENDPOINT: ENDPOINT의 값이 '옵션 값'일 경우 설정한 '인코딩 값'을 메시지에 할당한다. URL: URL 값이 '옵션 값'일 경우 설정한 '인코딩 값'을 메시지에 할당한다. OFFSET_LENGTH: OFFSET_LENGTH의 값이 '옵션 값'일 경우 설정한 '인코딩 값'을 메시지에 할당한다. <ul style="list-style-type: none"> Offset : 시작점 Length : 시작점부터의 길이
인코딩	설정할 인코딩을 입력한다.

• 메시지 타입 설정

메시지 타입 설정

☒ 메시지 타입 사용

[+ 추가](#)

• 파싱옵션 선택

OFFSET_LENGTH

• Offset

0

• Length

10

옵션 값
Option

[+ 추가](#)
[- 삭제](#)

[- 삭제](#)

요청 메시지

이름	메시지 아이디	메시지 타입 이름
Req_Header_R_Tutorial	Req_Header_R_Tutorial	Req_Header_R_TutorialFixedLength

정상 응답 메시지

이름	메시지 아이디	메시지 타입 이름
Req_Header_R_Tutorial	Req_Header_R_Tutorial	Req_Header_R_TutorialFixedLength

업무오류 응답 메시지

이름	메시지 아이디	메시지 타입 이름
Req_Header_R_Tutorial	Req_Header_R_Tutorial	Req_Header_R_TutorialFixedLength

메시지 타입 설정 화면

항목	설명
파싱옵션 선택	"인코딩 타입 설정"의 '파싱옵션 선택'과 동일하다. 각각의 옵션 값에 따라 선택한 메시지 타입으로 요청, 정상 응답, 업무오류 응답 메시지의 타입을 변경한다.

항목	설명
옵션값	파싱 옵션 선택에 따른 사용자 입력값이다.

• XML Self-Close 설정

XML Self-Close 설정

XML Self-Close 옵션의 기본 설정값은 TRUE 입니다.

☒ XML Self-Close 설정

[+ 추가](#)

• 파싱 옵션 선택

ENDPOINT

옵션 값	XML Self-Close Flag
HTTP_ADT_Tutorial.HTTP_EP_IN	TRUE

[+ 추가](#)
[- 삭제](#)

[- 삭제](#)

XML Self-Close 설정 화면

항목	설명
파싱 옵션 선택	"인코딩 타입 설정"의 '파싱옵션 선택'과 동일하다. 각각의 옵션 값에 따라 XML Self-Close 설정을 변경한다.
옵션값	파싱 옵션 선택에 따른 사용자 입력값이다.
XML Self-Close-Flag	XML Self-Close 여부를 설정한다.

2.2.5. XML 뷰어

거래/거래그룹은 .biztx 확장자를 지니는 파일로 저장되거나 XML 형식의 문서로 작성된다. 거래/거래그룹의 설정 정보는 파일의 XML 구조를 보며 동작 방식을 확인하고 점검할 수 있다. XML 뷰어는 XML을 조회하는 기능만 제공하고 편집 기능은 지원하지 않는다.

3. 메시지 에디터

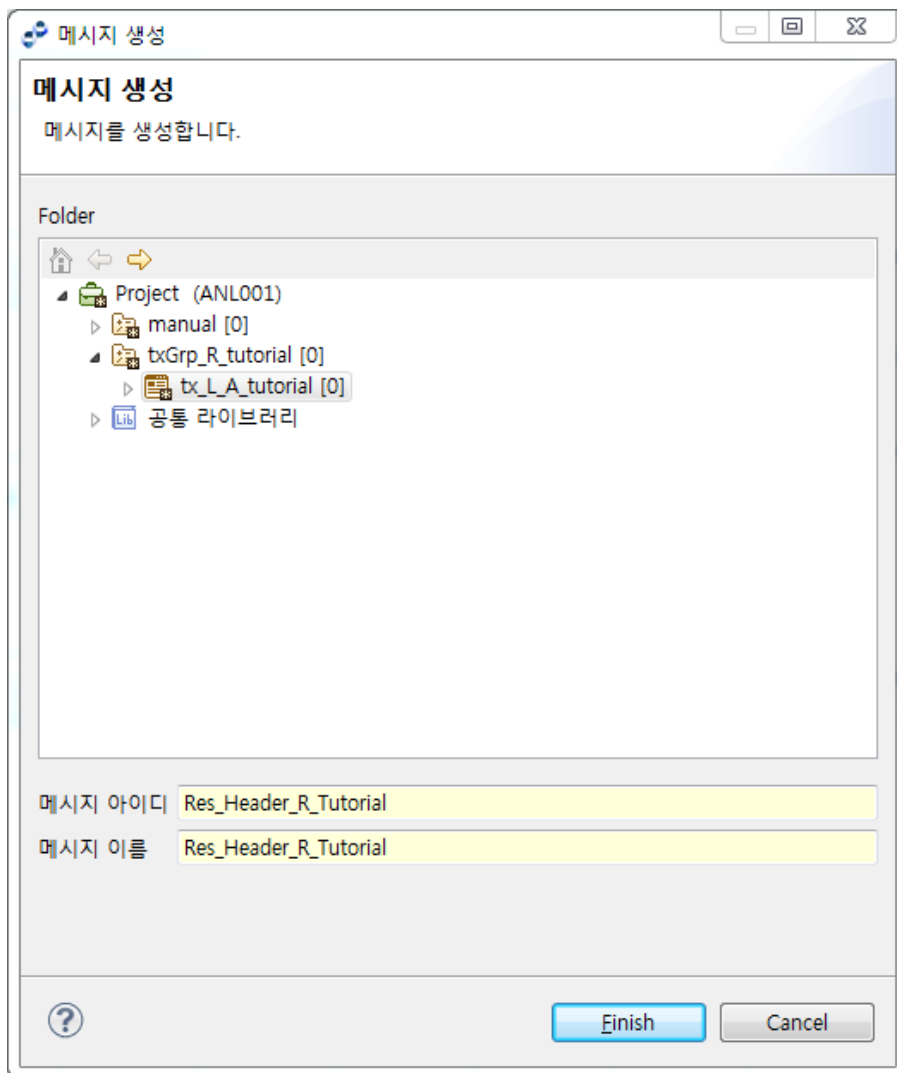
본 장에서는 AnyLink 스튜디오에 내장된 메시지 에디터의 기능과 사용법에 대해 설명한다.

3.1. 메시지 생성

Message(이하 메시지)는 전문을 추상화(abstraction)시킨 객체이다.

AnyLink의 어댑터는 메시지 객체를 이용하여 DataObject를 네트워크로 내보낼 바이트형 배열을 생성하고, 네트워크에서 수신한 바이트형 배열은 DataObject로 변환하여 플로우로 넘겨준다. 이때 DataObject를 바이트형 배열로 생성하는 과정을 Marshal이라고 하며, 반대로 바이트형 배열을 DataObject로 변화하는 과정을 Unmarshal이라고 한다.

메시지는 왼쪽 네비게이터의 **거래(그룹)** 리소스 컨텍스트 메뉴에서 **[새로만들기] > [메시지]**를 선택한다. **메시지 생성 다이얼로그**에서 생성할 **[거래/거래그룹]**을 선택한 후 항목을 입력하고 **[Finish]** 버튼을 클릭한다.



메시지 생성 다이얼로그

항목	설명
메시지 아이디	메시지의 아이디를 입력한다. 영어, 숫자, 특수문자(_)만 사용이 가능하며 첫 글자는 대문자로 입력한다.
메시지 이름	메시지의 이름을 입력한다. 한글, 영어, 숫자, 특수문자 입력이 가능하다. 메시지 이름은 XML Naming Convention을 따른다.

3.2. 메시지 에디터

메시지 리소스가 생성되면 **메시지 에디터 화면**으로 이동해서 해당 메시지의 정보를 관리할 수 있다.

메시지 에디터

• 메시지 정의

항목	설명
메시지 ID	메시지를 생성할 때 입력한 메시지 아이디이다.
메시지 이름	메시지를 생성할 때 입력한 메시지 이름이다.
설명	메시지에 대한 설명을 입력한다. 편의상 입력하는 것으로 입력하지 않아도 무방하다.

• 메시지 설정

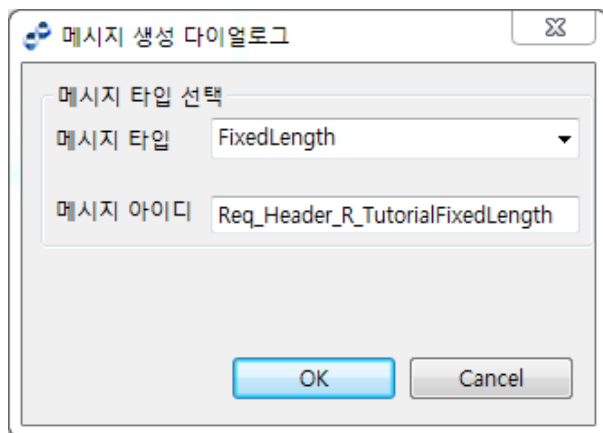
메시지 설정 영역은 메시지 목록과 메시지 타입별 설정 영역으로 구성된다. 각 메시지 유형별 설정에 대한 자세한 설명은 [메시지 설정](#)을 참고한다.

- 메시지 구성

메시지 구성 영역은 메시지 설정 영역에서 추가한 메시지를 구성하는 영역이다. 각 메시지 유형별 구성 항목이 달라진다. 메시지 구성에 대한 자세한 설명은 [메시지 구성](#)을 참고한다.

3.2.1. 메시지 설정

메시지 에디터 화면의 메시지 설정 영역은 메시지 목록과 메시지 타입별 설정 영역으로 구성된다. 메시지 목록의 [+추가] 버튼을 클릭해서 메시지를 생성한다. 메시지 생성 다이얼로그에서 각 항목을 입력한 후 [OK] 버튼을 클릭한다.



메시지 생성 다이얼 로그

다음은 입력항목에 대한 설명이다.

항목	설명
메시지 타입	<p>AnyLink에서는 다양한 메시지 타입을 선택한다.</p> <p>메시지 타입에 따라 메시지 타입별 설정에 항목이 달라진다.</p> <ul style="list-style-type: none"> ◦ Fixed Length ◦ Bitmap ◦ Dlimeter ◦ CSV ◦ XML ◦ JSON ◦ Value Object ◦ KeyValue ◦ FDL ◦ FML ◦ ISO8583
메시지 아이디	<p>메시지를 생성할 때 정의한 메시지 아이디에 메시지 타입이 추가되어 기본 생성된다.</p> <p>'메시지 타입'을 선택하면 '메시지 아이디'는 '리소스이름 + 메시지 타입'으로 생성된다. (수정 가능)</p>
Namespace 생성 안함	<p>체크할 경우 XML NameSapce를 생성 하지 않는다.</p> <p>'메시지 타입' 항목을 'XML'로 선택한 경우에만 설정 가능하다.</p>
Namespace 직접 입력	<p>체크할 경우 XML NameSapce를 자동으로 생성해주지 않고 직접 입력한다.</p> <p>'메시지 타입' 항목을 'XML'로 선택한 경우에만 설정 가능하다.</p>



메시지 필드 구성을 추가한 후 **메시지 목록**에서 추가해야 메시지를 생성시킬 수 있다.

메시지를 생성한 후 메시지를 설정한다.

메시지 목록 *

메시지 ID	메시지 타입
Res_Header_R_TutorialFixedLength	FixedLength

➕ 추가 ➖ 삭제 ↻ 수정

• Res_Header_R_TutorialFixedLength 메시지 타입 설정

Trim	none
Message Encoding	NONE
총 길이	10
공백 Include 필드 제거	<input type="checkbox"/>

메시지 에디터 - 메시지 설정

다음은 각 타입별 메시지 설정 항목에 대한 설명이다.

• 공통 항목

다음은 모든 메시지 타입에서 공통적으로 설정할 수 있는 항목이다.

항목	설명
Message Encoding	메시지의 인코딩 타입을 설정한다. AnyLink는 다음의 인코딩 타입을 지원한다. <ul style="list-style-type: none">• ASCII• EBCDIC• UTF-8• UTF-16• UTF-16BE• UTF-16LE• EUC-KR

• Fixed Length

항목	설명
Trim	메시지 Trim 여부를 설정한다. <ul style="list-style-type: none">• none : trim을 하지 않는다.• rtrim : 문자열의 오른쪽을 trim한다.• ltrim : 문자열의 왼쪽을 trim한다.• ltrim : 문자열의 양쪽을 trim한다.
총 길이	메시지 필드들의 Length 합을 보여준다. Include, Array 등의 설정으로 인해 총 길이의 합을 알 수 없는 경우 Unknown Length로 표시한다.
공백 Include 필드 제거	체크할 경우 배열데이터가 공백(0x20)이거나 null(0x00)일 경우 해당 필드는 객체로 생성하지 않는다.

• Bitmap

메시지 타입이 Bitmap인 경우 추가 설정 항목이 존재하지 않는다.

• Dlimeter

항목	설명
Field Delimiter	메시지 필드 구분 문자를 설정한다.

• CSV

메시지 타입이 CSV인 경우 추가 설정 항목이 존재하지 않는다.

• XML

항목	설명
Root Element Name	XML 메시지 Root Element Name을 설정한다.
Content Type	XML 메시지 Content Type을 설정한다.
Complex Type Name	XML 메시지 Complex Type Name을 설정한다.
Simple Type Name	Content Type이 Simple로 설정된 경우 XML 메시지 Simple Type Name을 설정한다.
UnmarshallException 무시	거래를 수행할 때 UnmarshallException이 발생하는 경우 UnmarshallException을 무시하고 거래를 진행할지 여부를 설정한다.
Namespace - Prefix 설정	Prefix를 변경할 경우 사용되는 옵션으로 변경하는 경우 Namespace와 Prefix를 입력한다.

• JSON

항목	설명
Null 처리 방식	메시지 필드값이 null 값인 경우 처리하는 방식을 설정한다. <ul style="list-style-type: none"> • NULL : null 유지 • EMPTY_STRING : 빈 문자열로 표시 • SKIP : 해당 태그 제거
UnmarshallException 무시	거래를 수행할 때 UnmarshallException이 발생하는 경우 UnmarshallException을 무시하고 거래를 진행할지 여부를 설정한다.
Object 이름	Object 이름을 입력한다. Json 객체의 Root 노드의 이름으로 Object 이름 하위 필드들로 메시지가 구성된다.

• Value Object

메시지 타입이 Value Object인 경우 추가 설정 항목이 존재하지 않는다.

• KeyValue

항목	설명
Delimiter	Delimiter로 사용할 값을 입력한다.
시작 Delimiter 제거	메시지의 앞에 Key-Value 메시지면 메시지 앞에 Delimiter를 붙이고, Key-Value 메시지가 아닐 경우에는 앞에 Delimiter를 생략한다.
Separator	Separator로 사용할 값을 입력한다.

항목	설명
Logical Name	<p>각 필드의 논리명을 입력한다.</p> <p>Java의 변수명 규약을 따르지 않고, 한글 입력도 가능하다.</p>
Field Type	<p>각 필드의 자료형을 선택한다.</p> <ul style="list-style-type: none"> • binary : Base64 방식으로 인코딩된 2진 데이터 타입 • BigDecimal : java.math에 포함되어 있는 BigDecimal 클래스 타입 • BigInteger : java.math에 포함되어 있는 BigInteger 클래스 타입 • bit : 0과 1만 값만 가지는 2진 데이터 타입 • bitmap : 0과 1만 표현 가능한 Map 형식의 2진 데이터 타입 • boolean : Java에서 사용하는 boolean 타입과 동일 • byte : Java에서 사용하는 Byte 타입과 동일 • char : Java에서 사용하는 char 타입과 동일 • Calendar : java.util에 포함되어 있는 Calendar 클래스 타입 • date : java.util에 포함되어 있는 Date 클래스 타입 • double : Java에서 사용하는 double 타입과 동일 • float : Java에서 사용하는 float 타입과 동일 • int : Java에서 사용하는 int 타입과 동일 • integer : java.lang에 포함되어 있는 Integer 클래스 타입 • include : 다른 메시지를 포함하는 경우 사용하는 타입 • long : Java에서 사용하는 long 타입과 동일 • number : java.lang에 포함되어 있는 Number 클래스 타입 • short : Java에서 사용하는 short 타입과 동일 • string : java.lang에 포함되어 있는 String 클래스 타입
Included Str. Name	<p>'Field Type' 항목이 'Include'일 경우, Include된 메시지의 이름과 위치를 보여준다. 직접 입력할 수는 없다.</p>
Included Str. Path	
Array Size	<p>메시지의 필드가 배열의 형태로 여러 개 입력되는 경우 배열의 개수를 넣는다. 입력하지 않으면 배열이 아니라고 간주한다.</p> <p>숫자를 입력하면 숫자가 배열의 크기가 되며, 같은 메시지 내의 필드명을 입력하면 해당 필드의 값을 배열의 크기로 설정할 수 있어서 가변배열 설정이 가능하다. 가변배열의 크기로서 설정하는 필드는 반드시 숫자 타입이어야 한다. 'unbounded'를 입력하는 경우 배열의 크기를 지정하지 않고 사용이 가능하다.</p>
Comment	<p>필드에 대한 설명을 입력한다. 입력하지 않아도 무방하다.</p>

항목	설명
Mask	<p>해당 메시지의 내용을 볼 때 마스크가 입력된 필드는 해당 마스크로 대체되어 보여진다.</p> <p>마스킹을 사용하기 위해서는 WebAdmin의 [운영관리] > [마스킹 설정]에서 업무시스템 또는 거래그룹/거래별로 마스킹 설정 정보를 설정해주어야 한다. 패스워드 같은 중요한 정보를 가리기 위해서 사용된다.</p> <ul style="list-style-type: none"> Mask 문자열이 한 글자인 경우 해당 문자로 원본 메시지를 변경한다. <p>예)</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>Mask 문자 : 'x' 원본 메시지 : 'abcd' ----- 결과 : 'xxxx'</p> </div> <ul style="list-style-type: none"> '?'의 경우 원본 메시지를 마스킹하지 않는다. <p>예)</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>Mask 문자 : '???-xxxx-???' 원본 메시지 : '010-1111-2222' ----- 결과 : '010-xxxx-2222'</p> </div> <ul style="list-style-type: none"> Mask 문자열로 '?'를 사용하는 경우 '?' 앞에 '\'를 추가한다. <p>예)</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>Mask 문자 : '?\??-xxxx-?\???' 원본 메시지 : '010-1111-2222' ----- 결과 : '0?0-xxxx-2?22'</p> </div> <p>[참고]</p> <p>'\ ' 문자의 경우 Java에서 Escape 문자로 구분되기 때문에 실제 입력할 때 '\\'로 입력한다.</p>
Corr. Field	<p>코릴레이션으로 사용되는 필드인지 설정한다.</p> <p>해당 항목을 'true'로 설정하면 MappingHandler에서 API를 통해 해당 항목의 값을 불러올 수 있다.</p> <p>메시지 필드의 타입을 설정했을 때만 설정할 수 있다.</p>

• 버튼

다음은 메시지 구성 영역에 존재하는 버튼에 대한 설명이다.

항목	설명
[추가]	[추가] 버튼을 클릭하면 필드를 추가할 수 있다.
[복수 필드 추가]	[복수 필드 추가] 버튼을 클릭하면 여러 개의 필드를 추가할 수 있다.
[삭제]	필드가 선택된 경우 활성화되며 선택된 필드를 삭제하는 경우 사용한다.
[검색]	필드 타입이 Include인 경우 활성화되며 버튼을 클릭할 때 Include할 메시지를 선택할 수 있는 다이얼로그가 생성된다.
[XML 변환]	XML 메시지를 초기화하는 경우 사용한다.
[VO Export]	메시지의 .java 파일과 .class 파일을 사용자가 지정한 경로에 Export한다. 각 파일명은 메시지 아이디와 동일하다.
[유효성검사]	서버에 등록된 메시지 유효성 검사 핸들러를 호출하여 각 필드의 유효성을 판단한다. 핸들러 생성 방법은 핸들러 를 참고한다.

메시지 타입별 메시지 구성 항목

다음은 메시지 타입별 메시지 구성의 항목에 대한 설명이다.

• Fixed Length

항목	설명
Length	'FixedLength' 타입에서 가장 중요한 설정이다. 메시지의 길이는 디폴트로 10이 지정되어 있으며 실제로 원하는 길이를 설정하기 위해서는 'Length' 항목을 변경해야 한다. 가변길이 설정을 위해서는 가변배열의 경우와 마찬가지로 숫자형 필드의 이름을 사용할 수 있다.
Offset	필드가 시작될 Offset을 나타내는 값으로 자동 계산되어 임의로 설정할 수 없다. 앞의 필드들의 Length를 모두 합친 값으로 계산되며, 앞의 필드 중에 Length가 동적으로 변하는 필드가 존재할 경우 Unknown으로 나타난다.
Align, Fill	DataObject 필드를 바이트형 배열로 Marshaling할 때 DataObject 필드 길이가 설정한 'Length'의 값보다 짧은 경우 좌/우를 맞추는(align) 후 나머지 위치에는 'Fill' 항목에 설정한 문자로 채워넣는다.

항목	설명
Decimal	<p>숫자 타입(Numeric type)인 경우 소수점 아래 몇 자리까지 보존할 것인지 설정한다.</p> <ul style="list-style-type: none"> • dto numeric 필드 값이 123.4321xxxx일 경우 이 필드에 해당하는 메시지 필드의 decimal =1이면 Marshal될 때 123.4로 Marshal되어 나간다. 즉, decimal=4이면 123.4321이 된다. • fixedLength 필드 길이가 6으로 지정되어 있다고 가정하고 decimal=4라고 한다. decimal=4이므로 소수점 4번째 자리까지 보존하면 123.4321이 되는데, 이 문자의 길이는 8(점(.) 포함)이므로 소수점 아래를 보존하도록 동작하여 3.4321로 Marshal된다. 즉, Decimal에 우선적으로 적용된다. • decimal=-1로 주는 경우 정수자리를 최대한 보존하고 남는 자리수만큼 소수점 자리를 찍는다. 위의 예에서 decimal=-1이라면, 123.43로 Marshal된다. 정수자리 123 보존, 그 후에 소수점 자리 찍을 수 있는 만큼 Marshal된다.
Point	<p>Decimal과 함께 사용하며 Mainframe에서 사용되는 가상소수점(또는 fixed point, 소수점의 위치가 약속되어 있는 형식)을 지원한다.</p> <p>예를 들어 fixed point 위치가 뒤에서 3번째라고 하면 decimal=3, point=false로 설정한다.</p>
Sign	<p>Mainframe의 가상 소수점을 사용하는 comp3, unpacked decimal encoding을 사용하는 필드에서 주로 사용된다. 인코딩 값에 따라 C, D, F의 값이 붙는다.</p> <ul style="list-style-type: none"> • sign = true이면 dto 필드 값의 음수 및 양수에 상관없이 무조건 sign을 붙인다. • sign = false이면 dto 필드 값이 음수(-)일 때만 sign을 붙인다.

항목	설명
Encode(FixedLength)	<p>메시지 필드의 개별 인코딩 타입이다.</p> <p>인코딩 종류는 다음과 같다.</p> <ul style="list-style-type: none"> • Char • KoreanSingleByte • KoreanDoubleByte • KoreanSpace • CharSOSI • CharSingleByte • AsciiSOSIInsert • NoConversion • COMP • COMP-2 • COMP-3 • UnpackedDecimal • BCD • IEEE754SinglePrecision • IEEE754DoublePrecision • Two's_Complement
Trailer	메시지 필드 타입이 'String'이거나 숫자형이고 인코딩 옵션이 BCD일 때, 최하위 4bit padding이 필요한 경우 Trailer 값으로 채워준다.
Date Pattern(yyyy-MM-dd)	메시지 필드 타입이 'Date'이거나 'Calendar'일 경우에 활성화되며, Date Format을 지정한다. (예: yyyyMMdd, yyyy-MM-dd, yy-MM-dd)
Endian	<p>레코드가 저장되는 머신의 레코드 저장 방식을 다음 중에 선택한다.</p> <ul style="list-style-type: none"> • Big Endian : 레코드의 순서와 동일하게 메모리에 저장하는 방식 • Little Endian : 레코드의 끝부터 메모리에 저장하는 방식
EOD	<p>EOD(End Of Data)는 맨 마지막 필드인 경우만 '참'이 되고, 나머지 필드는 '거짓'이 되어야 한다.</p> <p>EOD가 '참'인 경우 필드 값을 'Length'에 맞추어 자르지 않고 모두 바이트형 배열로 만든다.</p>
Unsigned	Unsigned 값인지 설정한다.

• Bitmap

항목	설명
Length	메시지의 길이는 디폴트로 10이 지정되어 있으며 실제로 원하는 길이를 설정하기 위해서는 'Length' 항목을 변경해야 한다. 가변길이 설정을 위해서는 가변배열의 경우와 마찬가지로 숫자형 필드의 이름을 사용할 수 있다.
EnCode(Bitmap)	메시지 필드의 개별 인코딩 타입이다.
BitMap NO.	각 필드에 고유의 Bitmap 번호를 할당한다. 여러 개의 필드 중 특정 필드들만을 사용할 경우에 필드의 인덱스 역할을 수행한다.
NO. fields in BitMap	FieldType이 Bitmap일 경우에만 입력할 수 있는 값으로 Bitmap으로 사용할 필드의 개수를 입력한다.

• Delimiter

항목	설명
Delimiter	메시지 필드가 Array로 설정된 경우 사용한다. 'a', 'b', 'c'처럼 직접 문자열을 입력하거나 '\n','\t' 같은 특수문자를 입력한다. 단, 인쇄할 수 없는 문자(Unprintable character)인 경우 '\u0000', '\u0020'(공백문자)와 같이 '\u' 유니코드 형태로 입력한다.
EOD	EOD(End Of Data)는 맨 마지막 필드인 경우만 '참'이 되고, 나머지 필드는 '거짓'이 되어야 한다.

• CSV

메시지 타입이 CSV인 경우 추가 설정 항목이 존재하지 않는다.

• XML

항목	설명
Is Element	해당 필드가 엘리먼트인지 속성인지 나타낸다. <ul style="list-style-type: none"> • true : 엘리먼트를 의미한다. • flase : 속성을 의미한다.
Attribute Name	이 이름을 입력한 필드는 속성으로 지정된다. 'Is Element'의 값이 'false'일 경우에만 유효하다.
Attr Use	필드가 속성일 경우에만 활성화되며 다음과 같은 옵션 중에서 선택할 수 있다. XSD에서의 'use' 속성과 동일한 역할을 한다. <ul style="list-style-type: none"> • optional : 해당 필드는 선택적으로 값을 할당할 수 있다. • required : 해당 필드에 무조건 값이 할당되어야 한다. • prohibited : 해당 필드에 값이 할당되어서는 안 된다.

항목	설명
Element Name	이 이름을 입력한 필드는 엘리먼트로 지정된다. 'Is Element' 의 값이 'true'일 경우에만 유효하다.
XML Type	선택된 필드의 타입에 해당되는 XML 타입을 QName 형식으로 표현한다. (예: {http://www.tmax.co.kr/test}tesType)
Min Occurs	XSD에서의 'minOccurs', 'maxOccurs' 속성과 동일한 역할을 한다.
Max Occurs	Max Occurs를 unbounded로 사용할 수도 있다.
Nillable	'true'일 경우 해당 필드에 null 값이 할당되어도 된다는 것을 의미한다.
Date Pattern(yyyy-MM-dd)	메시지 필드 타입이 'Date'이거나 'Calendar'일 경우에 활성화되며, Date Format을 지정한다.

• JSON

항목	설명
EnCode(JSON)	메시지 필드의 개별 인코딩 타입이다.
Date Pattern(yyyy-mm-dd)	메시지 필드 타입이 'Date'이거나 'Calendar'일 경우에 활성화되며, Date Format을 지정한다.
JsonKey	파싱할 JsonKey를 입력한다. 입력되어 있지 않을 경우 Physical Name으로 파싱한다.

• Value Object

메시지 타입이 Value Object인 경우 추가 설정 항목이 존재하지 않는다.

• KeyValue

항목	설명
Date Pattern(yyyy-mm-dd)	메시지 필드 타입이 'Date'이거나 'Calendar'일 경우에 활성화되며, Date Format을 지정한다.
Trim	메시지 Trim 여부를 설정한다. <ul style="list-style-type: none"> • none : trim을 사용하지 않는다. • rtrim : 필드의 오른쪽을 trim한다. • ltrim : 필드의 왼쪽을 trim한다. • ltrim : 필드의 양쪽을 trim한다.
Null Value	입력한 값과 실제 들어온 value가 일치할 경우 Marshal 시 값이 일치하는 key-value에 대해 포함하지 않고 전문을 생성한다. 값이 설정되어 있지 않거나 일치하지 않을 경우에는 포함하여 전문을 생성한다.

- **FDL(Fixed Definition Language)**

항목	설명
Field Key	<p>기본적인 필드 정보와 필드 키를 입력한다.</p> <p>AnyLink와 연동할 Tmax(FDL의 경우) 에 미리 등록되어 있는 필드 키 값을 사용해야 한다.</p>

- **FML(Fixed Manipulation Language)**

항목	설명
Field Key	<p>기본적인 필드 정보와 필드 키를 입력한다.</p> <p>AnyLink와 연동할 Tuxedo(FML의 경우)에 미리 등록되어 있는 필드 키 값을 사용해야 한다.</p>

- **ISO8583**

항목	설명
Length	<p>메시지의 길이는 디폴트로 10이 지정되어 있으며 실제로 원하는 길이를 설정하기 위해서는 'Length' 항목을 변경해야 한다.</p> <p>가변길이 설정을 위해서는 가변배열의 경우와 마찬가지로 숫자형 필드의 이름을 사용할 수 있다.</p>

4. 아웃바운드 룰 에디터

본 장에서는 AnyLink 스튜디오에 내장된 아웃바운드 룰 에디터의 기능과 사용법에 대해 설명한다.

4.1. 개요

아웃바운드 룰은 RTE로부터 아웃바운드 어댑터로 데이터를 전송하는 역할을 하는 서비스를 말하며 유용한 기능들을 제공한다. 아웃바운드 룰은 인바운드 어댑터로 전송된 메시지의 형식을 기억함은 물론 데이터의 전송이 성공적으로 끝난 경우와 실패한 경우에 원하는대로 메시지의 형식을 정의하고 내용을 변경할 수 있는 기능을 제공한다.

스튜디오에서 아웃바운드 룰을 생성하지 않을 경우 AnyLink는 가장 기본적인 기능을 제공하는 아웃바운드 룰을 생성하며, 만일 스튜디오에서 아웃바운드 룰을 생성한 경우 매칭되는 입력 데이터 형식의 메시지가 들어오면 자동으로 그 메시지를 매칭된 아웃바운드 룰이 처리하도록 동작한다.

원하는 경우 거래에 아웃바운드 룰을 설정시켜둘 수도 있으나 해당하는 경우에는 아웃바운드 룰에서 입력 메시지와 아웃바운드 룰 메시지 간의 데이터 매핑을 해야 하는 작업이 추가적으로 요구될 수도 있다.



1. 본 장에서는 커스텀 로그 아웃바운드 룰, 배치 아웃바운드 룰에 대해서만 설명한다. 기타 아웃바운드 룰 생성에 필요한 자세한 내용은 각 어댑터별 사용자 안내서를 참고한다.
2. 프로젝트와 거래/거래그룹이 미리 생성되어 있지 않은 경우 [프로젝트 생성](#), [거래/거래그룹 생성](#)을 참고하여 먼저 프로젝트 및 거래/거래그룹을 생성해야 한다.

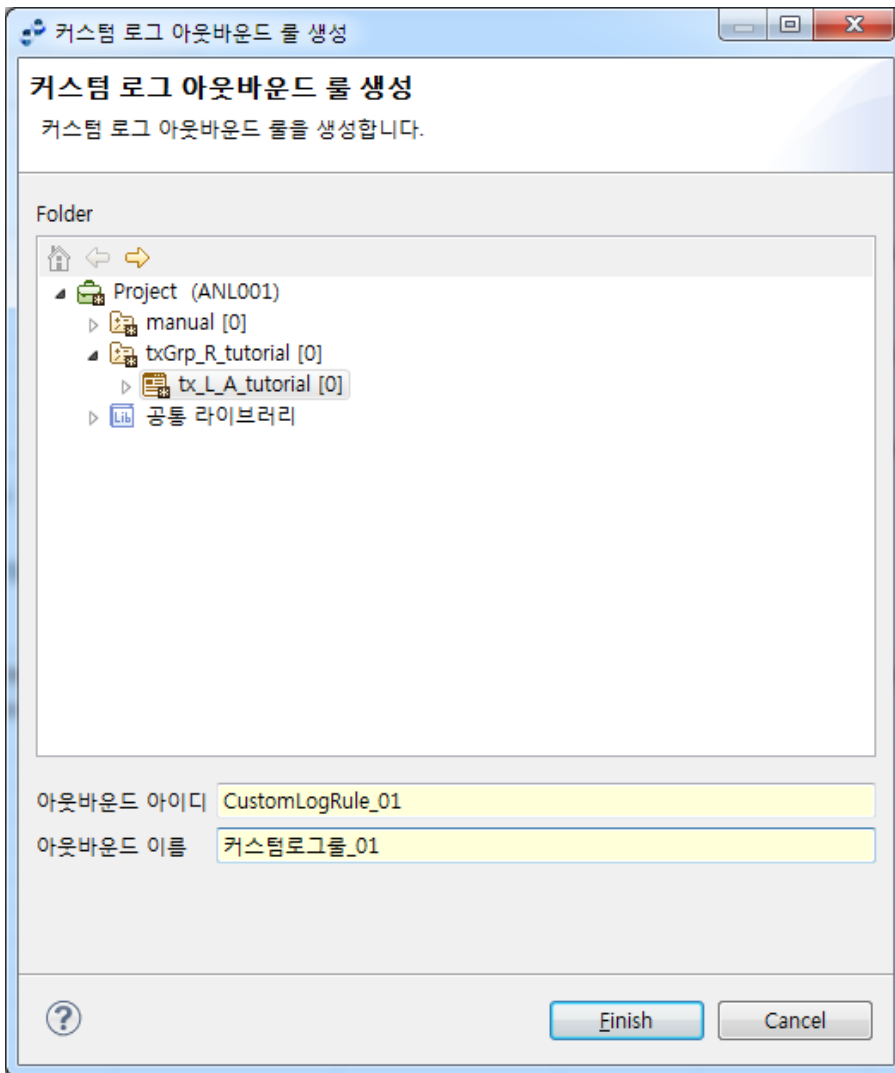
4.2. 커스텀 로그 아웃바운드 룰

커스텀 로그 아웃바운드 룰은 서비스 플로우의 룰 액티비티, 응답 메시지 액티비티의 요청, 응답 부분과 Start 메시지 이벤트에 매핑을 통해 설정이 가능하다. 본 절에서는 커스텀 로그 아웃바운드 룰의 생성과 설정을 설명한다.

4.2.1. 커스텀 로그 아웃바운드 룰 생성

아웃바운드 룰 생성은 왼쪽 프로젝트 네비게이터의 컨텍스트 메뉴에서 **[새로만들기] > [아웃바운드 룰] > [커스텀 로그 아웃바운드 룰]**을 선택한다.

아웃바운드 룰 생성 화면에서 각 항목을 설정하고 **[Finish]** 버튼을 클릭한다.



커스텀 로그 아웃바운드 룰 생성 화면

항목	설명
아웃바운드 아이디	아웃바운드 룰 아이디를 입력한다. 영어, 숫자, 특수문자(_)만 입력이 가능하며 첫 글자는 영어로 입력한다.
아웃바운드 이름	아웃바운드 룰 이름을 입력한다. 한글, 영어, 숫자, 특수문자(-, _) 입력이 가능하다. 아웃바운드 룰 이름은 XML Naming Convention을 따른다.

4.2.2. 커스텀 로그 아웃바운드 룰 설정

다음은 커스텀 로그 아웃바운드 룰 설정에 대한 설명이다.

- 아웃바운드 룰 정의

아웃바운드 룰 정의

- 프로토콜
- 아웃바운드 룰 ID *
- 아웃바운드 룰 이름 *
- 요청처리 타임아웃(ms) :
- Endpoint * (Group)

- 설명
- 암호화 사용 ☐
- 유저 클래스 선택
- 메시지 Clone ☐
- View 사용 ☐

아웃바운드 룰 정의

항목	설명
프로토콜	아웃바운드 룰이 속하는 프로토콜로 아웃바운드 룰 항목을 만들 때 값이 고정된다.
아웃바운드 룰 ID	아웃바운드 룰의 아이디를 입력한다. 아이디는 중복을 허용하지 않으므로 다른 아웃바운드 룰 아이디의 겹치지 않도록 유의하도록 한다. (필수 입력 항목)
아웃바운드 룰 이름	아웃바운드 룰의 이름을 입력한다. (필수 입력 항목)
요청처리 타임아웃(ms)	아웃바운드 방향으로 데이터를 전송할 때 데이터 전송을 요청한 이후 최대한 기다릴 수 있는 시간을 설정한다.
Endpoint(Group)	데이터를 전송할 외부 시스템과 연결된 엔드포인트 또는 엔드포인트 그룹을 설정한다. [검색] 버튼을 클릭할 경우 리소스 검색 화면을 통해 원하는 엔드포인트나 엔드포인트 그룹을 선택할 수 있다. 해당 설정은 필수로 설정해야 한다.
설명	아웃바운드 룰에 대한 설명을 입력한다. 각 아웃바운드를 구별하기 쉽게 도와주므로 보다 상세하게 적을 것을 권장한다.
암복호화 사용	로그 내용의 암호화 여부를 선택한다.
유저 클래스 선택	암복호화에 사용할 유저 클래스를 선택한다. [검색] 버튼을 클릭할 경우 리소스 검색 화면을 통해 원하는 유저 클래스를 선택할 수 있다.
메시지 Clone	메모리 효율은 떨어질 수 있으나 로깅을 위해 변수를 복제하여 거래 수행 시점에 영향을 받지 않고 변수의 값을 로깅할 수 있는 옵션이다.
View 사용	데이터베이스의 View를 사용한다. View를 사용할 시 요청메시지, 로그 테이블의 매핑, Insert SQL 문의 편집이 불가능하다.

• 요청 메시지

요청 메시지는 입력으로 들어오는 메시지의 형식을 의미한다. 아웃바운드 엔드포인트에서 입력으로 들어온 메시지의 형식을 맞추고, 매칭하기 위해 사용한다.

▼ 요청 메시지

이름	메시지 ID	타입 ID	배열 선택
요청 입력헤더	ReqInHeader	ReqInHeaderFixedLength	설정 안함
요청 입력바디	ReqInBody	ReqInBodyFixedLength	설정 안함

가져오기

+ 추가

- 삭제

요청 메시지

• 로그 테이블 설정

◦ 로그 테이블 이름

엔드포인트에 포함된 로그 테이블을 선택한다. 로그 테이블 이름을 설정하기 위해선 상단의 엔드포인트를 필수로 설정해야 한다. **[매핑]** 버튼을 클릭하면 **매핑 다이얼로그**([\[로그 테이블 설정\] - \[커스텀 로그 테이블 정보\] - 매핑 다이얼로그](#))에서 요청 메시지와 로그 테이블의 컬럼 이름의 매핑이 가능하다.

로그 테이블 이름

AL_CUSTOM_TABLE

매핑

[로그 테이블 설정] - 로그 테이블 이름

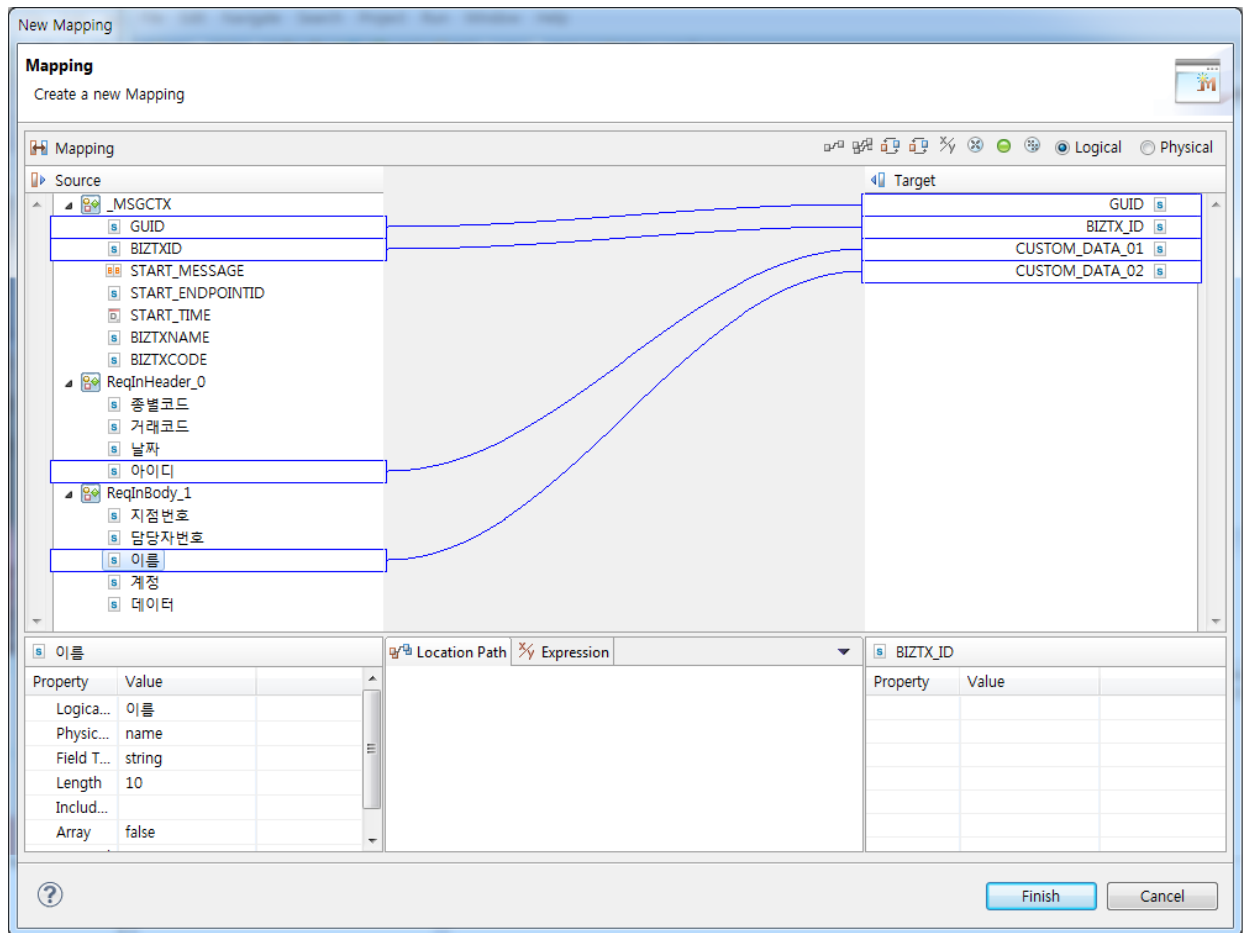
엔드포인트에 설정된 DataSource의 DB 계정이 DML 권한만 부여되어서 테이블 목록을 가져올 수 없을 경우 JEUS DAS(DomainAdminServer)의 jvm 옵션을 다음과 같이 설정하면 테이블 목록을 가져올 수 있다.

-Danylink.dis.db.noperm=true

◦ 매핑 다이얼로그

AnyLink에서 제공하는 거래의 트랜잭션/트레이스 정보나 요청메시지로 설정한 메시지를 커스텀로그 테이블의 컬럼과 매핑하여 로깅시에 매핑된 정보들을 DB 테이블에 저장한다.

Source 영역에는 AnyLink에서 제공하는 트랜잭션/트레이스 정보를 매핑할 수 있는 **_MSGCTX** 메시지와 커스텀 로그 룰에 설정한 요청 메시지가 나타난다. Target 부에는 로그 테이블 이름에서 설정한 테이블의 DB 컬럼들이 표시된다.



[로그 테이블 설정] - [커스텀 로그 테이블 정보] - 매핑 다이얼로그

◦ 커스텀 로그 테이블 정보

선택한 커스텀 로그 테이블의 정보가 나타난다. [추가], [삭제] 버튼을 클릭해서 테이블을 변경할 수 있다.

커스텀 로그 테이블 정보

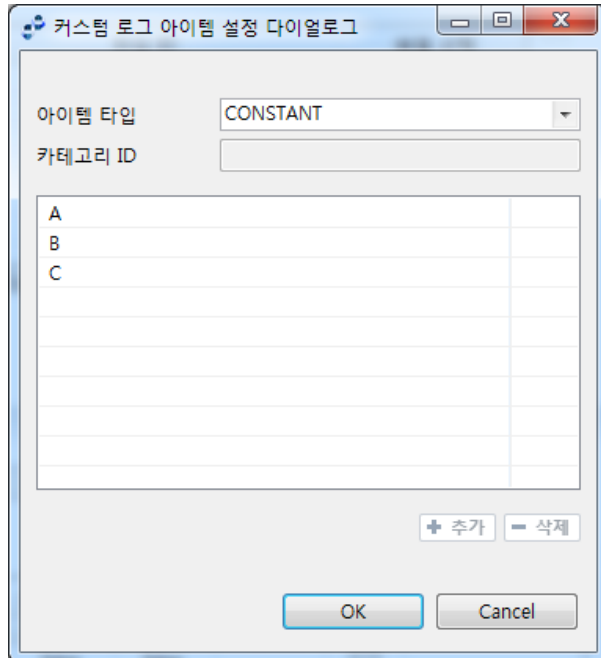
컬럼 Physical Name	컬럼 Logical Name	데이터 타입	검색 조건	조회 조건	정렬 조건	아이템
GUID	GUID	VARCHAR	false	false		...
BIZTX_ID	BIZTX_ID	VARCHAR	false	false		...
CUSTOM_DATA_01	CUSTOM_DATA_01	VARCHAR	false	false		...
CUSTOM_DATA_02	CUSTOM_DATA_02	VARCHAR	false	false		...

[로그 테이블 설정] - 커스텀 로그 테이블 정보

항목	설명
컬럼 Physical Name	컬럼의 Physical Name을 입력한다.
컬럼 Logical Name	컬럼의 Logical Name을 입력한다.

항목	설명
데이터 타입	컬럼의 데이터 타입을 나타내며 변경이 가능하다. 데이터 타입이 BLOB, CLOB인 경우 검색 조건, 조회 조건이 false로 설정되면 변경할 수 없다.
검색 조건	WebAdmin에서 해당 컬럼을 검색할지 여부를 선택한다. 해당 조건을 true로 변경하지 않으면 커스텀로그 조회 화면의 검색 조건으로 출력되지 않는다.
조회 조건	WebAdmin에서 검색된 결과에 대하여 해당 컬럼을 조회할지 여부를 선택한다.
정렬 조건	WebAdmin에서 검색된 결과에 대하여 정렬하는 방식을 선택한다. <ul style="list-style-type: none"> ◦ 오름차순 : 오름차순으로 정렬한다. ◦ 내림차순 : 내림차순으로 정렬한다.
아이템	WebAdmin에서 조회하는 경우 콤보박스 리스트로 나타날 아이템들을 커스텀하여 설정할 수 있다. 버튼을 클릭하면 커스텀 로그 아이템 설정 다이얼로그 ([로그 테이블 설정] - 커스텀 로그 아이템 설정 다이얼로그)가 나타나고, 해당 다이얼로그에서 아이템을 설정할 수 있다.

다음의 화면에서 아이템을 설정한 후 **[OK]** 버튼을 클릭한다.



[로그 테이블 설정] - 커스텀 로그 아이템 설정 다이얼로그

항목	설명
아이템 타입	<ul style="list-style-type: none"> ◦ CONSTANT : 하위 테이블에 고정 값을 입력한다. ◦ USER_META : WebAdmin의 메타정보를 이용한다.

항목	설명
카테고리 ID	아이템 타입이 USER_META일 경우 아이템 리스트로 사용할 메타정보의 카테고리 ID를 입력한다.
CONSTANT 아이템 테이블	아이템 타입이 CONSTANT일 경우 아이템 리스트로 사용할 값들을 [+추가] 버튼을 이용하여 추가하고, [-삭제] 버튼을 이용해 삭제한다.

◦ INSERT SQL 문

요청 메시지와 로그 테이블의 컬럼 이름의 매핑에 의해 생성된 SQL 문이 나타난다. **[편집]** 버튼을 클릭하면 SQL 문 편집이 가능하고 **[리셋]** 버튼을 클릭하면 편집 이전의 SQL 문으로 돌아간다.

INSERT SQL 문

```
INSERT INTO AL_CUSTOM_TABLE ( BIZTX_ID, CUSTOM_DATA_01, CUSTOM_DATA_02, GUID )
VALUES ( :/_MSGCTX.BIZTXID, :/ReqInHeader_0.id, :/ReqInBody_1.name, :/_MSGCTX.GUID )
```

편집

리셋

[로그 테이블 설정] - INSERT SQL 문

4.2.3. 커스텀 로그 예제

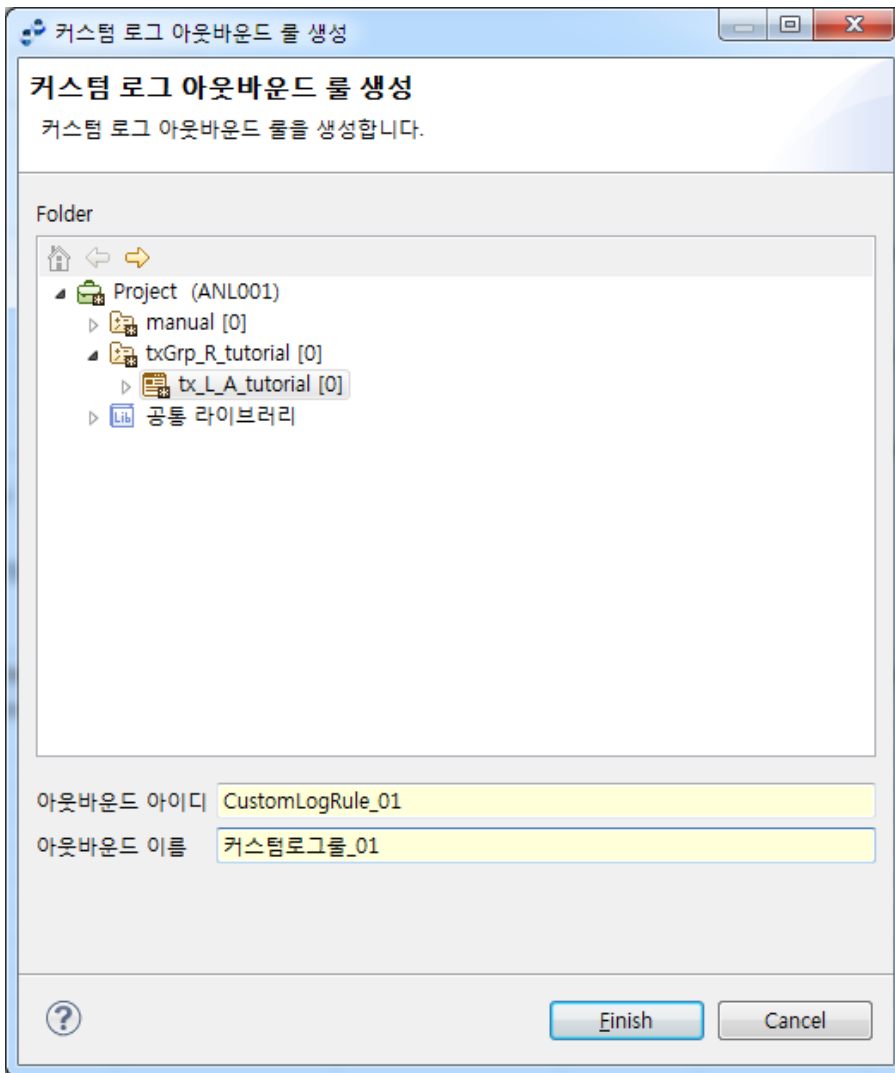
본 절에서는 거래 수행 중 커스텀 로깅을 하는 예제를 설명한다.



거래 수행에 필요한 리소스 생성, 배포 및 거래 수행 방법은 "TCP 어댑터 사용자 안내서"를 참고한다.

커스텀 로그 아웃바운드 룰 생성

네비게이터에서 거래/거래그룹 컨텍스트 메뉴에서 **[새로만들기]** > **[아웃바운드 룰]** > **[커스텀 로그 아웃바운드 룰]**을 선택한다.



커스텀 로그 아웃바운드 룰 생성 화면

항목	설정
아웃바운드 아이디	CustomLogRule_01
아웃바운드 이름	커스텀로그룰_01

커스텀 로그 아웃바운드 룰 설정

• 아웃 바운드 룰 정의

아웃바운드 룰 정의

• 프로토콜	LOG	• 설명	
• 아웃바운드 룰 ID *	CustomLogRule_01	• 암호화 사용	<input type="checkbox"/>
• 아웃바운드 룰 이름 *	커스텀로그룰_01	• 유저 클래스 선택	<input type="text"/> 검색
• 요청처리 타임아웃(ms) :	10,000	• 메시지 Clone	<input type="checkbox"/>
• Endpoint * (Group)	default-log-endpoint	• View 사용	<input type="checkbox"/>

아웃바운드 룰 정의

항목	설정
프로토콜	LOG
아웃바운드 룰 ID	CustomLogRule_01
아웃바운드 룰 이름	커스텀로그룰_01
요청처리 타임아웃(ms)	10000
Endpoint(Group)	default-log-endpoint

• 요청 메시지

▼ 요청 메시지

이름	메시지 ID	타입 ID	배열 선택
요청입력헤더	ReqInHeader	ReqInHeaderFixedLength	설정 안함
요청입력바디	ReqInBody	ReqInBodyFixedLength	설정 안함

가져오기

+

추가

-

삭제

요청 메시지

항목	설정
요청 메시지	요청입력헤더(ReqInHeader)
	요청입력바디(ReqInBody)

• 로그 테이블 설정

▼ 로그 테이블 설정

로그 테이블 이름

AL_CUSTOM_TABLE

매핑

커스텀 로그 테이블 정보

컬럼 Physical Name	컬럼 Logical Name	데이터 타입	검색 조건	조회 조건	정렬
GUID	GUID	VARCHAR	true	true	
BIZTX_ID	BIZTX_ID	VARCHAR	true	true	
CUSTOM_DATA_01	CUSTOM_DATA_01	VARCHAR	true	true	
CUSTOM_DATA_02	CUSTOM_DATA_02	VARCHAR	true	true	

INSERT SQL 문

INSERT INTO AL_CUSTOM_TABLE (BIZTX_ID, CUSTOM_DATA_01, CUSTOM_DATA_02, GUID)
VALUES (:/_MSGCTX.BIZTXID, :/ReqInHeader_0.id, :/ReqInBody_1.name, :/_MSGCTX.GUID)

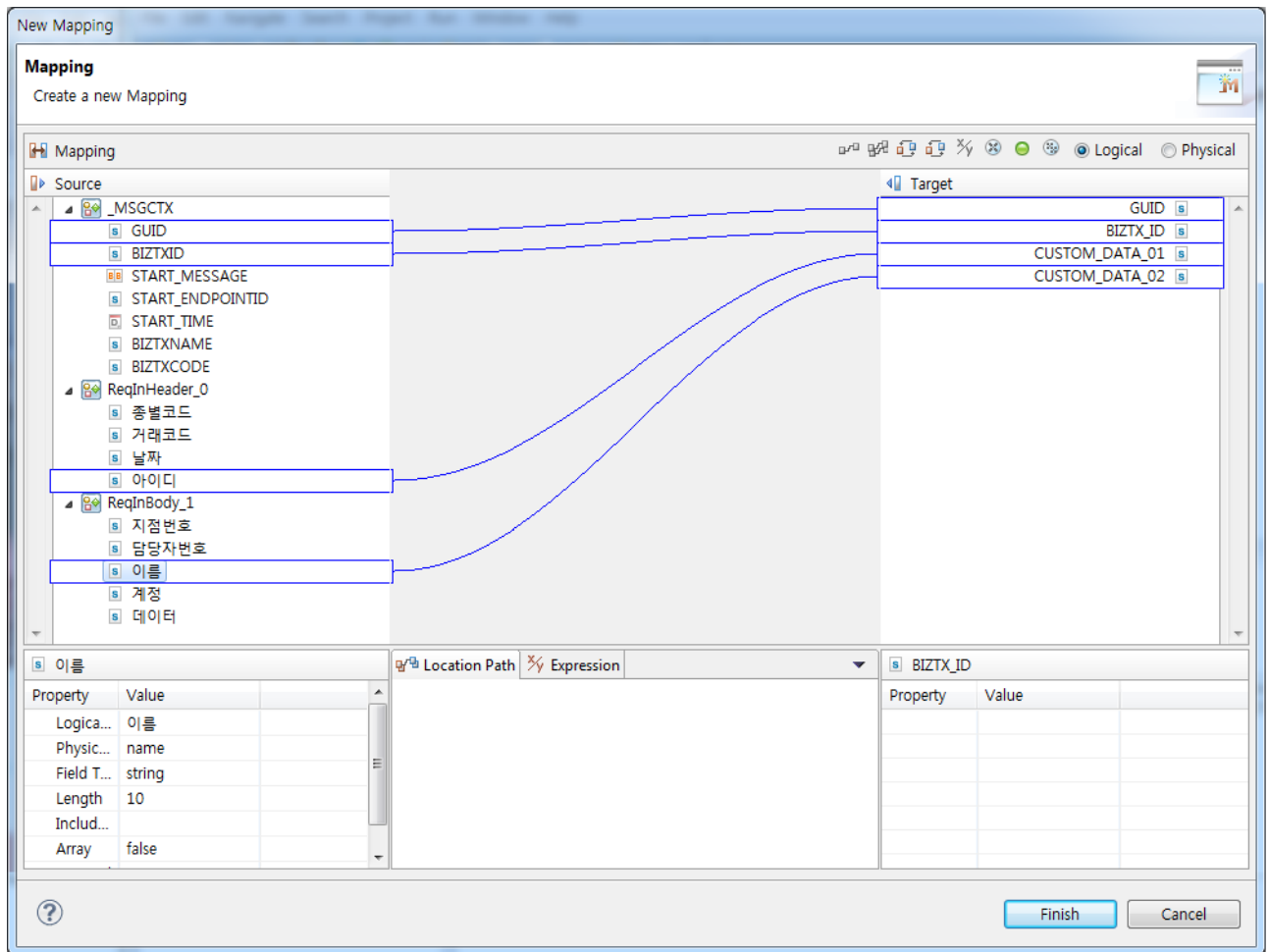
편집

리셋

로그 테이블 설정

항목	설정
로그 테이블 이름	AL_CUSTOM_TABLE

항목	설정
커스텀 로그 테이블 정보	검색 조건 : true
	조희 조건 : true

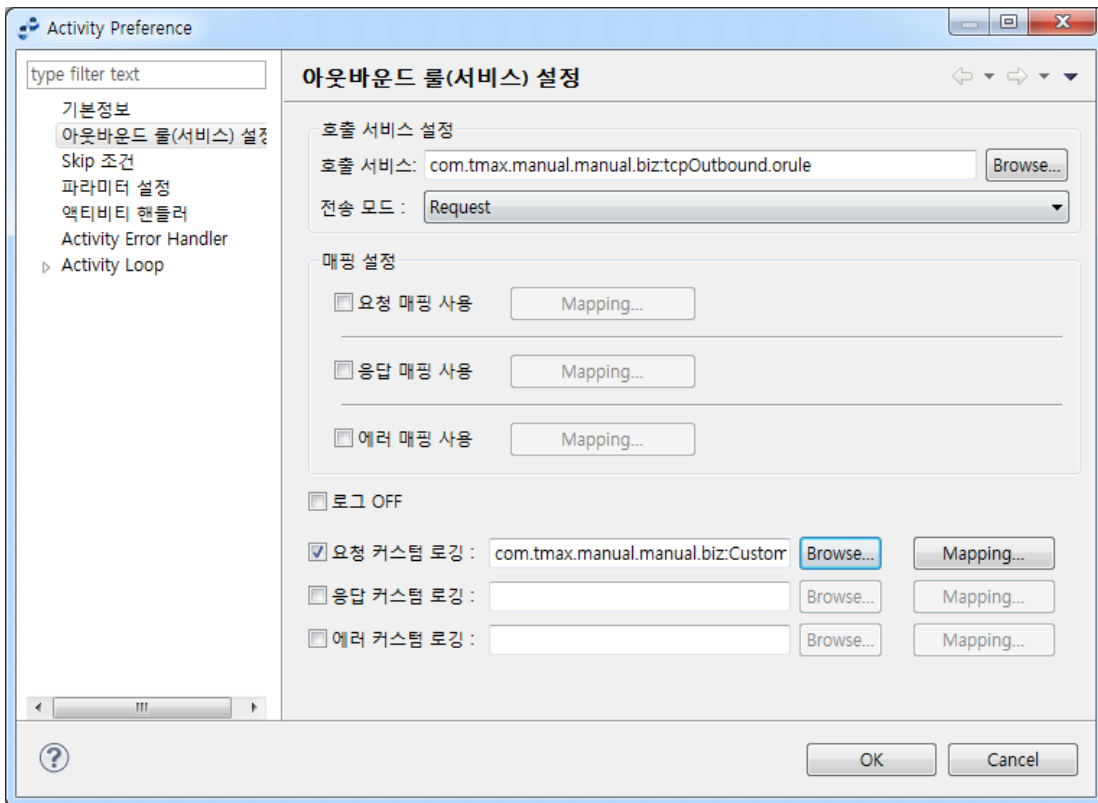


로그 테이블 매핑

플로우 설정

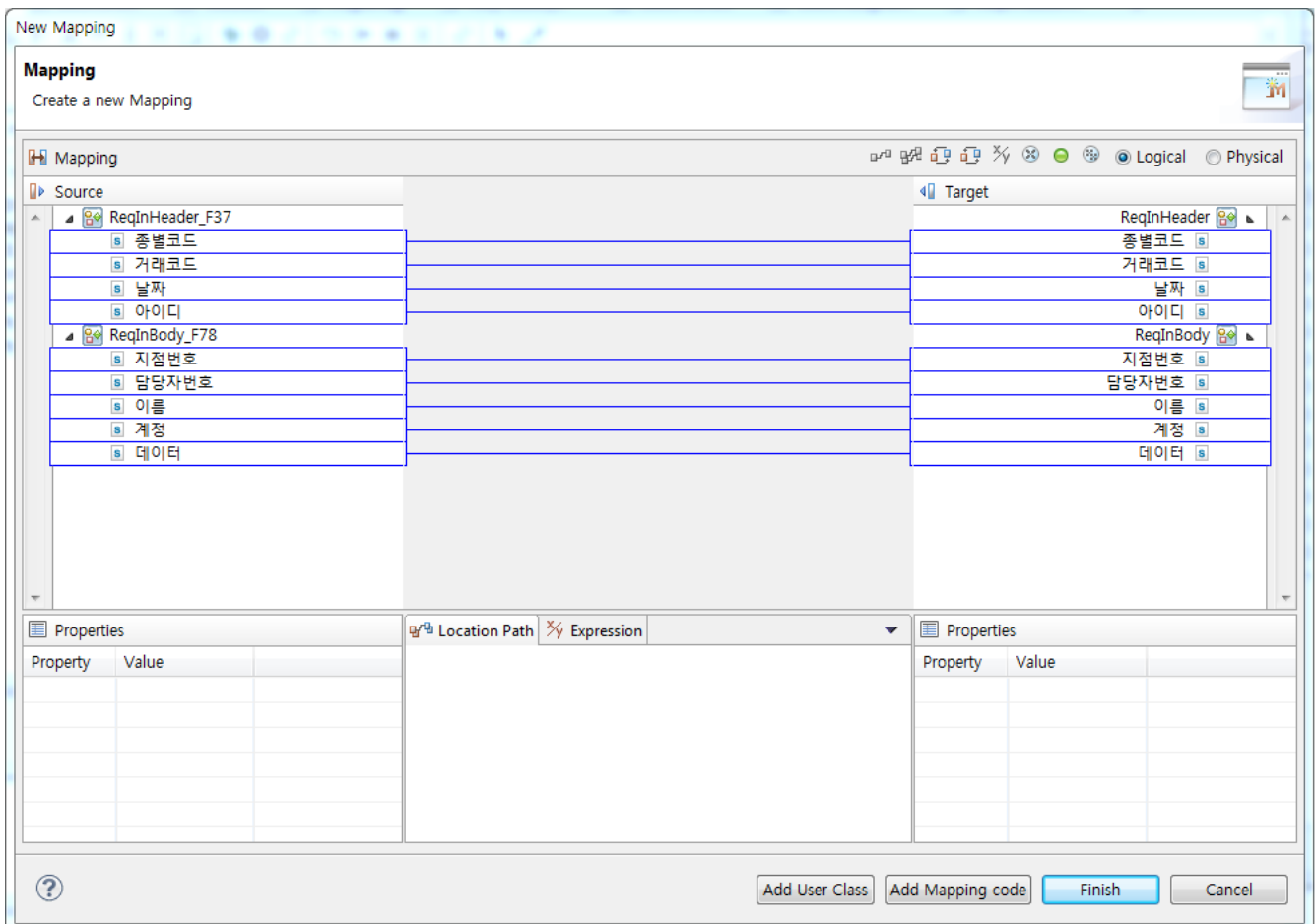
본 예제에서는 TCP 아웃바운드 룰의 요청 커스텀 로깅(TCP 아웃바운드 룰에 입력되는 메시지에 대한 커스텀 로깅) 설정 방법을 설명한다.

다음은 **Activity Preference** 화면의 **[아웃바운드 룰(서비스) 설정]** 메뉴를 선택한 화면이다. 요청 커스텀 로깅을 선택하고 **[Browse...]** 버튼을 클릭하여 커스텀 로그 아웃바운드 룰 파일(CustomLogRule_01.orule)을 선택한다.



아웃바운드 호출 - 아웃바운드 룰(서비스) 설정

다음은 요청 커스텀 로그 매핑 화면이다. **Source**에는 액티비티의 Input 파라미터가 설정되고 **Target**에는 커스텀 로그 아웃바운드 룰의 요청 메시지가 설정된다.



요청 커스텀 로그 매핑

거래 테스트 수행 결과

커스텀 로깅 결과는 WebAmin의 [모니터링] > [커스텀 로그] 메뉴에서 확인할 수 있다.

커스텀 로그

manual > biz > CustomLogRule_01

GUID

BIZTX_ID

CUSTOM_DATA_01

☒ 포함 검색

CUSTOM_DATA_02

GUID	BIZTX_ID	CUSTOM_DATA_01	CUSTOM_DATA_02
COA80F10JBEMFAECNFDGFFJ000000A0	A001	데이터01	데이터02
COA80F10JBFLDKAAKJCFJJ00000193	A002	Data01	Data02

Export

페이지당 목록 수: 50

커스텀 로그 확인 페이지

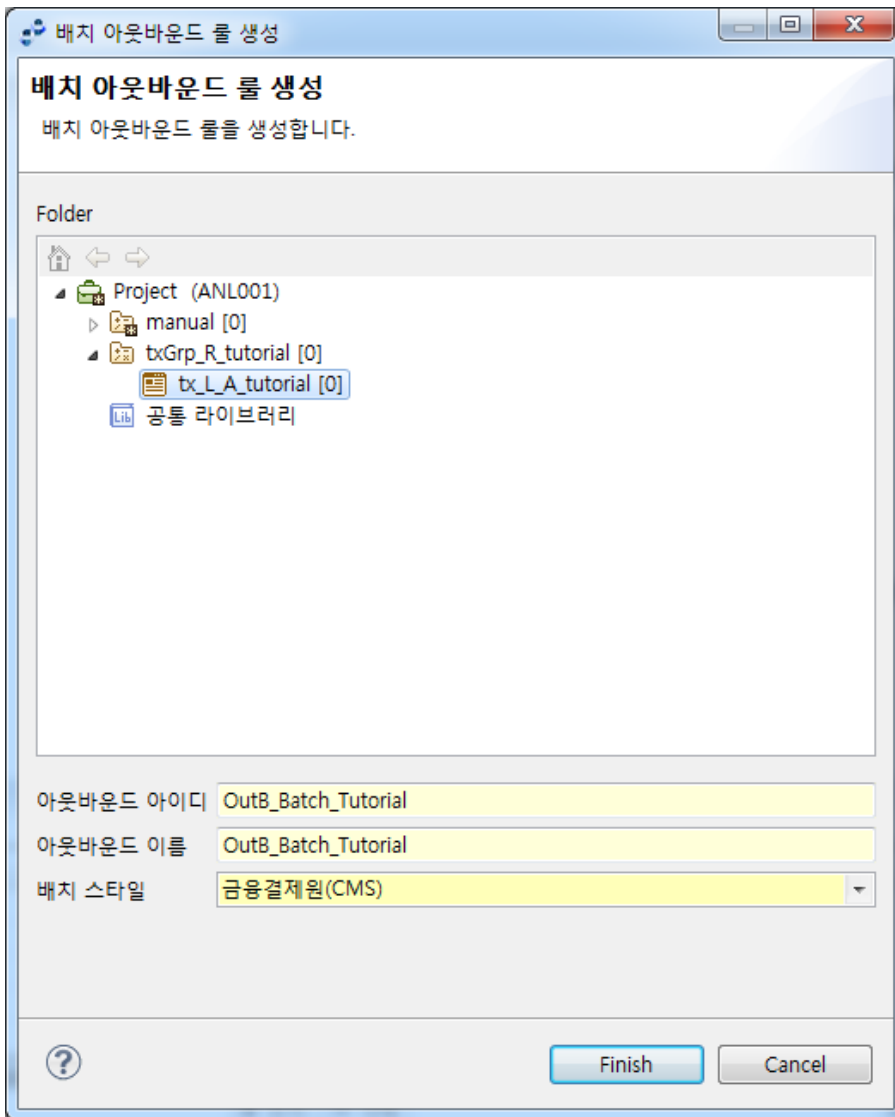
항목	설정
GUID	수행된 거래의 GUID
BIZTX_ID	TCP 액티비티의 Input 파라미터인 ReqInHeader 메시지의 '거래코드' 필드 값
CUSTOM_DATA_01	TCP 액티비티의 Input 파라미터인 ReqInHeader 메시지의 '아이디' 필드 값
CUSTOM_DATA_02	TCP 액티비티의 Input 파라미터인 ReqInHeader 메시지의 '이름' 필드 값

4.3. 배치 아웃바운드 룰

본 절에서는 배치 아웃바운드 룰의 생성과 설정에 대한 부분을 설명한다. 배치 아웃바운드 룰의 경우 스펙에 맞게 필요한 정보들을 스타일 형태로 제공하는 것으로 각 스타일 별로 제공하는 설정들이 다르다.

4.3.1. 배치 아웃바운드 룰 생성

아웃바운드 룰 생성은 왼쪽 프로젝트 네비게이터의 컨텍스트 메뉴에서 [새로만들기] > [아웃바운드 룰] > [배치 아웃바운드 룰]을 선택한다. 배치 아웃바운드 룰 생성 화면에서 각 항목을 설정하고 [Finish] 버튼을 클릭한다.



배치 아웃바운드 룰 생성 화면

항목	설명
아웃바운드 아이디	아웃바운드 룰 아이디를 입력한다. 영어, 숫자, 특수문자(_)만 입력이 가능하며 첫 글자는 영어로 입력한다.
아웃바운드 이름	아웃바운드 룰 이름을 입력한다. 한글, 영어, 숫자, 특수문자(.,_) 입력이 가능하다. 아웃바운드 룰 이름은 XML Naming Convention을 따른다.

항목	설명
배치 스타일	<p>배치 스타일을 선택한다.</p> <p>AnyLink에서 제공하는 배치 스타일은 다음과 같다.</p> <ul style="list-style-type: none"> ◦ 금융결제원(CMS) : 금융결제원 CMS 스타일은 0600 계열의 배치 스타일이다. ◦ 데이콤_EDI(DACOM_EDI) : DACOM EDI 스타일은 DACOM MagicLink에서 제공하는 EDI 서비스에 호스트로 접속하려는 경우에 사용되는 일괄전송 스타일이다. ◦ 보험개발원(KIDI) : 보험개발원 스타일은 의무보험가입관리전산망과 보험사/공제조합 시스템 간의 파일 송수신을 수행하기 위한 0600 계열의 배치 스타일이다. ◦ 삼성네트웍스(SSNW) : 삼성 네트웍스 스타일은 삼성 네트워크의 Alien 게이트웨이와 통신할 수 있는 기능이다. ◦ 생명보험협회(KLIA) : 생명보험협회 스타일은 아래 은행연합회신용정보 스타일과 유사한 스타일로, 수신 개시 요청으로 파일 송신 후 업무 종료 요청을 하지 않고 수신 개시를 요청한 기관으로부터 업무 종료 요청을 수신 대기하는 것이 은행연합회신용정보 스타일과 유일하게 다른 점이다. ◦ 손해보험 공통(SONBO) : 방카슈랑스 손해보험공통 스타일이다. 손해보험 공통(손보공통) 배치 스타일은 방카슈랑스 시스템에서 은행과 보험사 사이의 파일 송수신을 위한 스타일이다. ◦ 손해보험협회(KNIA) : 손해보험협회 전산망 접속을 위한 데이터 송수신 스타일로 송/수신 데이터전문의 전체 길이를 알 수 없는 점이 손해보험공통 스타일과 다르다. ◦ 은행연합회신용정보(KFB) : 은행연합회 신용정보 스타일은 은행연합회(이하 센터)에서 제안한 일괄전송 프로토콜로, 참가기관(은행)에서 센터로 신용정보 파일을 송신 또는 수신할 때 사용된다. ◦ 파일전송(FTP) : FTP(File Transfer Protocol)는 TCP/IP 기반의 서버/클라이언트 간 파일 전송을 위한 프로토콜이다. ◦ 한국신용정보원(KCREDIT) : 한국신용정보원에서 제안한 일괄전송 프로토콜로, 참가기관에서 한국신용정보원으로 신용정보 파일을 송신 또는 수신할 때 사용된다. ◦ 한국예탁결제원(KSD) : 한국예탁결제원시스템(KSD시스템)과 참가기관 시스템 간 발생하는 데이터 및 파일 송수신을 위한 스타일이다. ◦ 한국정책금융공사(KOTA) : 한국정책금융공사 스타일은 금융결제원과 유사한 스타일로 한국정책금융공사와 통신할 수 있는 스타일이다.



0600 계열은 금융결제원 일괄전송, 은행연합회, CMS, 수납장표, 한신평 스타일 등의 업무가 이에 해당한다. 이 배치 스타일들은 모두 0600/001이 개시전문이고 결번처리 과정이 동일하다. 0600/001 전문이 개시 전문으로 들어오면 헤더를 주고받거나 파일 송수신지시전문 등을 주고받으면 0320 전문으로 데이터를 보내준다. 0320 전문을 연속해서 최대 100건까지 전송한 후 0620 전문으로 결번 요청, 0300으로 결번 응답을 주면 결번이 있을 경우 0310으로

결번 데이터를 전송하고, 결번이 없으면 0320 전문을 계속해서 보낸다. 이와 같은 구조로 되어 있는 배치 스타일을 의미한다.

4.3.2. 배치 아웃바운드 룰 설정

다음은 배치 아웃바운드 룰 설정 항목에 대한 설명이다.

• 아웃바운드 룰 정의


아웃바운드 룰 정의			
• 프로토콜	<input type="text" value="BATCH"/>	• 설명	<input type="text"/>
• 아웃바운드 룰 ID *	<input type="text" value="OutB_Batch_Tutorial"/>	• 배치 진행률 사용	<input type="text" value="false"/>
• 아웃바운드 룰 이름 *	<input type="text" value="OutB_Batch_Tutorial"/>	• 송수신구분 코드	<input type="text"/>




아웃바운드 룰 정의

항목	설명
프로토콜	아웃바운드 룰이 속하는 프로토콜로 아웃바운드 룰 항목을 만들 때 값이 고정된다.
아웃바운드 룰 ID	아웃바운드 룰의 아이디를 입력한다. 아이디는 중복을 허용하지 않으므로 다른 아웃바운드 아이디의 겹치지 않도록 유의하도록 한다. (필수 입력 항목)
아웃바운드 룰 이름	아웃바운드 룰의 이름을 입력한다. (필수 입력 항목)
설명	아웃바운드 룰에 대한 설명을 입력한다. 각 아웃바운드를 구별하기 쉽게 도와주므로 보다 상세하게 적을 것을 권장한다.
배치 진행률 사용	WebAdmin의 배치 진행률 메뉴를 사용하기 위해서는 해당 옵션을 true로 설정한다. (기본값: false)
송수신구분 코드	해당 배치가 송신 배치인지 수신 배치인지 설정한다. 배치 진행률 사용이 true일 경우에만 설정가능하다.

• 요청 메시지

입력으로 들어오는 메시지의 형식을 의미한다. 배치 스타일 정보를 조회할 수 있는 파일코드, txDate, Filler 등의 정보를 포함한 메시지를 설정한다.

요청 메시지			
이름	메시지 ID	타입 ID	배열 선택
 Req_Header...	Req_Header_R_Tutorial	Req_Header_R_TutorialFixedLength	설정 안함


 가져오기  추가  삭제

요청 메시지

• 응답 메시지

배치 스타일 정보를 조회한 정보를 매핑할 메시지를 설정한다. **[추가]** 버튼을 클릭해서 응답 메시지를 설정할 수 있다.

▼ 응답 메시지

이름	메시지 ID	타입 ID	배열 선택
 Res_Header...	Res_Header_R_Tutorial	Res_Header_R_TutorialFixedLength	설정 안함

가져오기

+ 추가


- 삭제

응답 메시지

• 파일리스트

[Input Mapping] 버튼은 파일 코드에 요청 메시지를 매핑하고 **[Output Mapping]** 버튼은 응답 메시지를 스타일 항목에 매핑한다.

금융결제원(CMS) 파일 리스트

배치 파일	설명
ABCD	파일코드1
EFGH	파일코드2
 CODE	대표파일

파일코드

+ 추가 - 삭제

대표파일

+ 추가 - 삭제

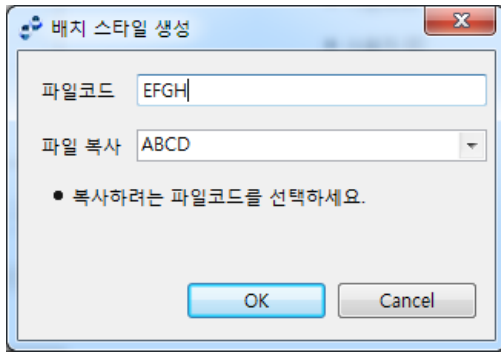
매핑 설정

InputMapping OutputMapping

파일 리스트

◦ 파일코드

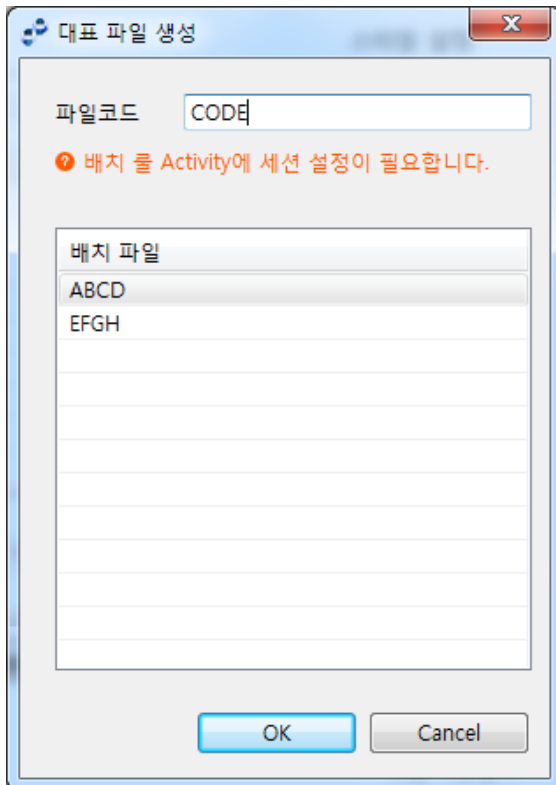
[추가] 버튼을 클릭해서 다이얼로그가 생성되면 추가할 파일 코드를 입력하고 추가된 파일 코드에 대한 스타일 설정이 가능하다. 파일 복사에서 기 생성된 파일 코드를 선택하는 경우 선택된 스타일과 동일한 스타일을 갖는 스타일이 생성된다. **[삭제]** 버튼을 클릭하면 파일 코드 제거가 가능하다.



배치 스타일 생성 다이얼로그

◦ 대표 파일

대표 파일은 파일 코드들의 묶음에 대한 코드를 의미한다. 대표 파일 **[추가]** 버튼을 클릭해서 다이얼로그가 생성되면 대표 파일을 설정하고 대표 파일이 포함할 파일 코드들을 선택한다. **[삭제]** 버튼을 클릭하면 대표 파일 제거가 가능하다.

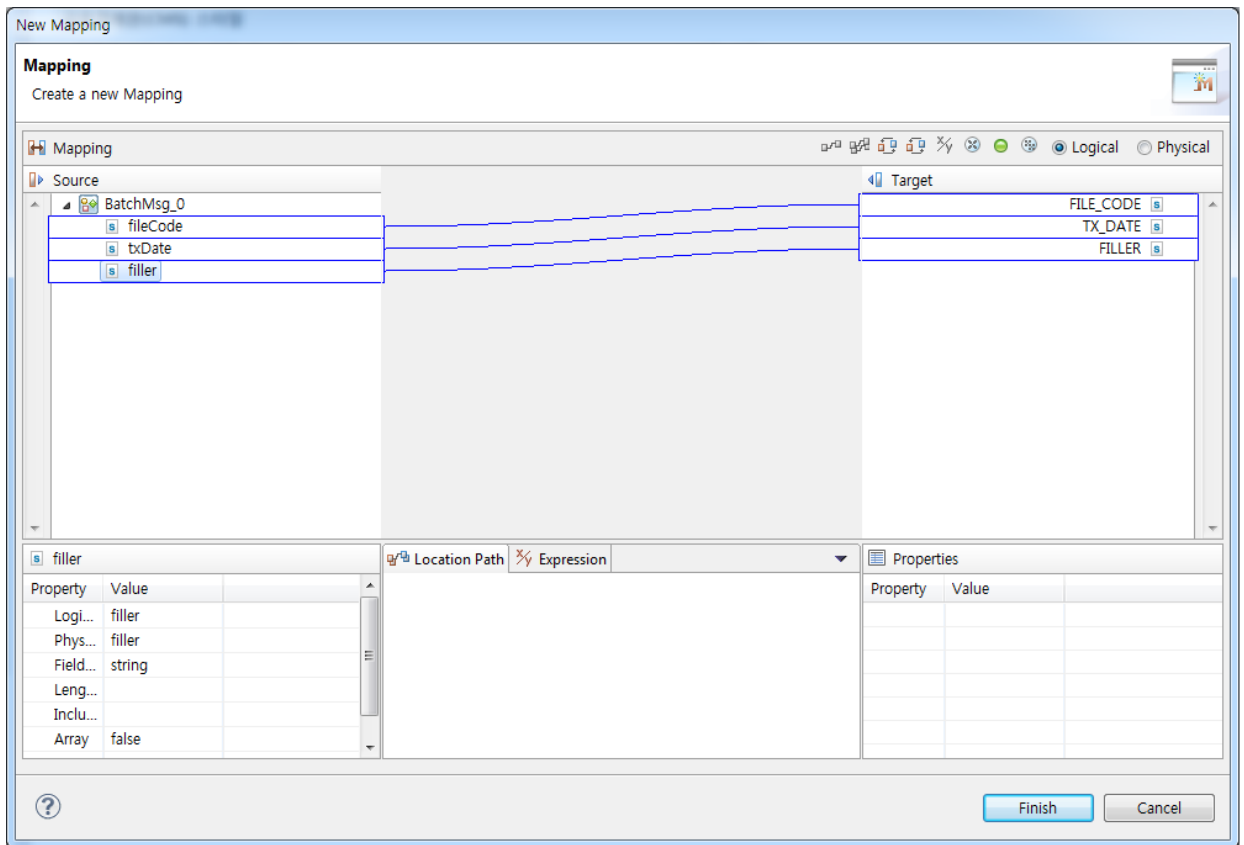


대표 파일 생성 다이얼로그

◦ Input Mapping

배치 스타일에 설정한 정보를 가져오기 위한 파일코드, txDate, Filler 등의 정보를 매핑한다.

Source 영역에는 배치 룰에 설정한 요청 메시지가 배치되고, Target 영역에는 FILE_CODE, TX_DATE, FILLER가 배치된다.



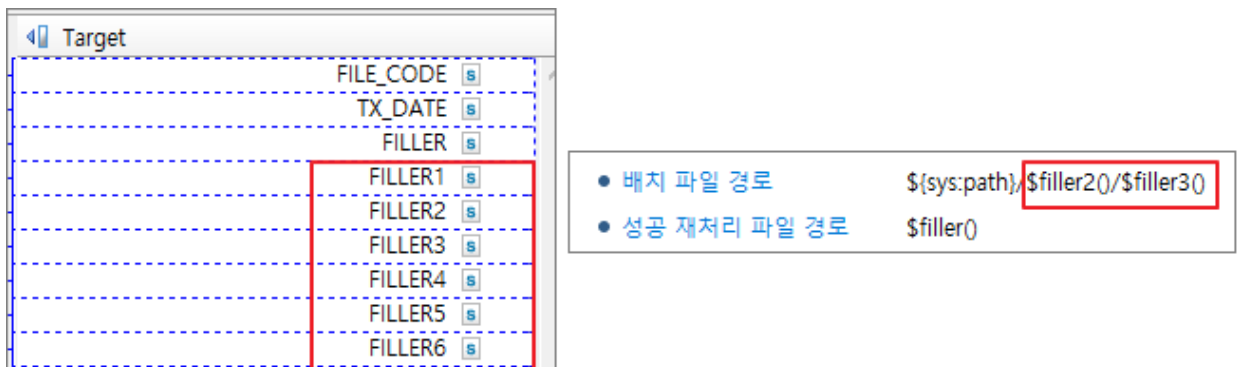
Input Mapping 다이얼로그

◦ Input Mapping Filler

[2023.02 stable 버전 기능추가]

배치 룰 > 배치 스타일 > 매핑설정: InputMapping > Target 화면에 FILLER1~6 필드를 추가했다.

- 배치를 > 배치정보에서 추가된 FILLER1~6 값을 함수로 이용할 수 있다.
 - \$filler1() ~ \$filler6()로 사용한다.
 - 기존 FILLER 필드는 동일하게 \$filler()로 사용한다.
 - 모니터링 > 배치 진행률 화면에서 출력을 확인한다. (경로, 파일명)

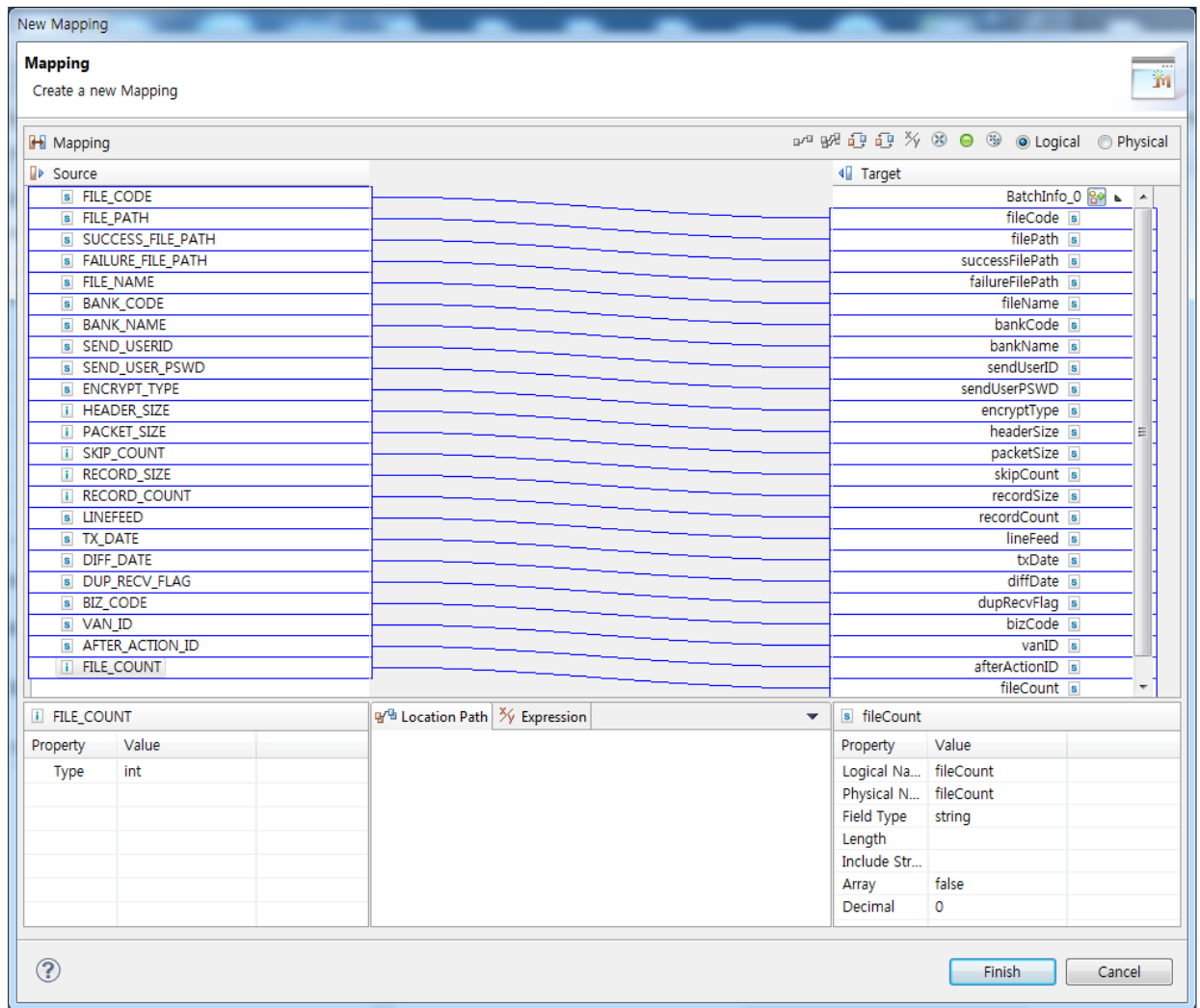


Input Mapping Filler

◦ Output Mapping

배치 스타일 정보를 응답 메시지에 매핑한다. Input Mapping을 통해서 요청한 배치에 대한 스타일 정보를 응답 메시지로 매핑하여 사용할 수 있다.

Source 영역에는 스타일 설정에서 설정한 값들과, AFTER_ACTION_ID, FILE_COUNT가 배치된다.



Output Mapping 다이얼로그

배치 스타일 명세

배치 스타일에 대한 명세를 작성할 수 있다.

배치 스타일 명세

배치 스타일 명세

스타일 설정

배치 스타일은 배치 업무를 수월하게 작성하기 위하여 플로우 내에서 공통으로 사용되는 값들을 파일 코드별로 제공한다. 실제 스펙 문서와 동일하지 않을 수 있다.

◦ 금융결제원 CMS 스타일

스타일 설정

- 파일코드
- 설명
- 배치 파일 경로
- 성공 재처리 파일 경로
- 실패 재처리 파일 경로
- 파일명
- 기관코드
- 기관명
- 사용자 ID
- 사용자 PW
- 암호화유형
- 헤더크기
- 데이터 패킷크기
- 결번검증건수
- 레코드 크기
- 레코드 갯수
- 개행문자
- 파일 기준일자
- 영업일 달력
- 파일 기준일자 변동분
- 중복수신 여부
- 업무코드
- VAN ID
- 후처리 ID

금융결제원 CMS 스타일

항목	설명
파일코드	파일 코드를 입력한다. (필수 입력 사항)
설명	파일 코드에 대한 설명을 입력한다. 파일 리스트의 설명 컬럼에 출력된다.
배치파일 경로	배치 파일이 위치할 해당 노드 내 경로를 입력한다.
성공 재처리 파일 경로	성공 재처리를 위한 해당 노드 내 경로를 입력한다.
실패 재처리 파일 경로	실패 재처리를 위한 해당 노드 내 경로를 입력한다.

항목	설명
파일명	파일명을 입력한다.
기관코드	해당 기관의 코드를 입력한다.
기관명	해당 기관의 이름을 입력한다.
사용자 ID	사용자 아이디를 입력한다.
사용자 패스워드	패스워드를 입력한다.
암호화유형	암호화를 사용할 경우 암호화 유형을 선택한다. <ul style="list-style-type: none"> ◦ 사용안함 ◦ 금융결제원 CMS 알고리즘 ◦ 금융결제원 CMS 알고리즘 (2009) ◦ 금융결제원 CMS 알고리즘 (2015)
헤더크기	데이터전문에서 파일 내역을 제외한 전문 길이를 등록한다. 파일정보전문을 보낼 때 전문 Byte 수 항목을 설정하는 데 사용된다.
데이터 패킷크기	전문 최대 크기에서 공통 부분의 ' 헤더크기 '를 뺀 길이를 등록한다.
결번검증건수	입력 단위로 결번검증을 수행한다.
레코드 크기	해당 파일에 정의된 데이터 레코드의 Byte 수를 입력한다.
레코드 갯수	파일을 보낼 때 한 시퀀스에 들어갈 레코드 개수를 입력한다.
개행문자	개행문자로 사용할 문자를 선택한다.
파일 기준일자	전송 기준일자로 스케줄러에 전문 등록할 경우 당일이 아닌 날짜의 데이터를 송수신할 수 있다.
영업일 달력	WebAdmin에서 생성한 영업일 달력을 설정하여 파일 기준일자 변동분 계산시 휴일을 제외하여 계산한다.
파일 기준일자 변동분	스케줄 등록을 통해 파일을 송수신할 때 파일 기준일자를 기준으로 며칠 전 일자 파일을 보낼지 등록한다. <ul style="list-style-type: none"> ◦ 음수(-) : 이전 날짜가 설정된다. ◦ 양수(+) : 이후 날짜가 설정된다. ◦ 사용하지 않을 경우 : 파일 기준일자 날짜가 설정된다.
중복수신 여부	중복 수신 여부를 선택한다.
업무코드	업무코드를 입력한다.
VAN ID	VAN ID를 입력한다.
후처리 ID	후처리 ID를 입력한다.

◦ 데이콤 EDI(DACOM EDI) 스타일

데이콤 EDI에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• 헤더크기	42
• 데이터 패킷크기	4054
• 파일 기준일자	%m%d
• 영업일 달력	CalID 🔍 검색
• 파일 기준일자 변동분	-1
• 개행문자	설정 안함 ▼
• 레코드 크기	100
• 사용자 ID	USER_ID
• 사용자 PW	PSWD
• JOB TYPE	TYPE1
• 상대방 ID	USER_ID2
• 업무코드	CODE
• VAN ID	
• 후처리 ID	
• 기관코드	
• 기관명	

데이콤 EDI

항목	설명
JOB TYPE	잡의 타입을 입력한다.
상대방 ID	상대방의 ID를 입력한다.

◦ 보험개발원(KIDI) 스타일

보험개발원에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• SYSTEM ID	SYSTEM_ID
• 기관코드	CODE
• 기관명	Tmax
• 사용자 ID	USER_ID
• 사용자 PW	PSWD
• 헤더크기	42
• 데이터 패킷크기	4054
• 결번검증건수	5
• 레코드 크기	100
• 레코드 갯수	2
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID <input type="button" value="Q 검색"/>
• 파일 기준일자 변동분	-1
• 중복수신 여부	Yes

보험개발원

항목	설명
SYSTEM ID	시스템 ID를 입력한다.

◦ 삼성 네트워크(SSNW) 스타일

삼성 네트워크에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• 기관코드	CODE
• 기관명	Tmax
• 사용자 ID	USER_ID
• 사용자 PW	PSWD
• 기관 ID	ID
• 기관 PW	PW
• 결번검증건수	5
• 레코드 크기	100
• 레코드 갯수	2
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID Q 검색
• 파일 기준일자 변동분	-1
• 중복수신 여부	Yes

삼성 네트워크

항목	설명
기관 ID	기관의 ID를 입력한다.
기관 PW	기관의 패스워드를 입력한다.

◦ 생명보험협회(KLIA) 스타일

생명보험협회에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	<input type="text" value="ABCD"/>
• 설명	<input type="text" value="파일코드1"/>
• 배치 파일 경로	<input type="text" value="REPOSITORY/ftp/folder"/>
• 성공 재처리 파일 경로	<input type="text" value="REPOSITORY/ftp/folder2"/>
• 실패 재처리 파일 경로	<input type="text" value="REPOSITORY/ftp/folder3"/>
• 파일명	<input type="text" value="AAAAAA.txt"/>
• TRANSACTION CODE	<input type="text" value="TCODE"/>
• SYSTEM ID	<input type="text" value="SYSTEM_ID"/>
• 송신기관코드	<input type="text" value="SCODE"/>
• 수신기관코드	<input type="text" value="RCODE"/>
• 단말기 ID	<input type="text" value="TERMINAL_ID"/>
• 단말기 조작자 ID	<input type="text" value="USER_ID"/>
• 헤더크기	<input type="text" value="42"/>
• 데이터 패킷크기	<input type="text" value="4054"/>
• 결번검증건수	<input type="text" value="5"/>
• 레코드 크기	<input type="text" value="100"/>
• 레코드 갯수	<input type="text" value="2"/>
• 개행문자	<input type="text" value="설정 안함"/>
• 파일 기준일자	<input type="text" value="%m%d"/>
• 영업일 달력	<input type="text" value="CalID"/> <input type="button" value="Q 검색"/>
• 파일 기준일자 변동분	<input type="text" value="-1"/>
• 중복수신 여부	<input type="text" value="Yes"/>

생명보험협회

항목	설명
TRANSACTION CODE	트랜잭션 코드를 입력한다.
SYSTEM ID	시스템 ID를 입력한다.
송신기관코드	송신기관코드를 입력한다.
수신기관코드	수신기관코드를 입력한다.
단말기 ID	단말기 ID를 입력한다.
단말기 조작자 ID	단말기 조작자 ID를 입력한다.

◦ 방카슈랑스 손해보험공통(SONBO) 스타일

손해보험공통 스타일에서 입력하는 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• 기관코드	CODE
• 기관명	Tmax
• 헤더크기	42
• 데이터 패킷크기	4054
• 레코드 크기	100
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID 🔍 검색
• 파일 기준일자 변동분	-1
• 업무코드	
• VAN ID	
• 후처리 ID	

방카슈랑스 손해보험공통 스타일

◦ 손해보험협회(KNIA) 스타일

손해보험협회에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

손해보험협회 스타일은 송/수신 데이터 전문의 전체 길이를 알 수 없기 때문에 WebAdmin TCP 엔드포인트 설정에서 메시지/장애처리 탭의 길이정보 키 사용유무를 체크하고, 길이 값에 표현식을 사용해서 길이 정보를 입력해주어야 한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• TRANSACTION CODE	TCODE
• SYSTEM ID	SYSTEM_ID
• 기관코드	CODE
• 기관명	Tmax
• 사용자 ID	USER_ID
• 사용자 PW	PSWD
• 헤더크기	42
• 데이터 패킷크기	4054
• 결번검증건수	5
• 레코드 크기	100
• 레코드 갯수	2
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID 🔍 검색
• 파일 기준일자 변동분	-1
• 중복수신 여부	Yes

손해보험협회

항목	설명
TRANSACTION CODE	트랜잭션 코드를 입력한다.
SYSTEM ID	시스템 ID를 입력한다.

◦ 은행연합회 신용정보(KFB) 스타일

은행연합회 신용정보에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• 기관코드	CODE
• 기관명	Tmax
• 사용자 ID	USER_ID
• 사용자 PW	PSWD
• 암호화유형	사용 안함
• 헤더크기	42
• 데이터 패킷크기	4054
• 결번검증건수	5
• 레코드 크기	100
• 레코드 갯수	2
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID 🔍 검색
• 파일 기준일자 변동분	-1
• 중복수신 여부	Yes
• 업무코드	
• VAN ID	
• 후처리 ID	

은행연합회 신용정보

◦ 파일전송 스타일

파일전송 스타일에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

● 파일 코드	ABCD
● 설명	파일코드1
● 소스 파일 경로	REPOSITORY/ftp/Source
● 소스 파일 백업 경로	REPOSITORY/ftp/Source/BackUp
● 로컬 파일 경로	REPOSITORY/ftp/Local
● 로컬 파일 백업 경로	REPOSITORY/ftp/Local/BackUp
● 타겟 파일 경로	REPOSITORY/ftp/Target
● 소스 파일명	Source.txt
● 로컬 파일명	Local.txt
● 타겟 파일명	Target.txt
● 기관코드	CODE
● 기관명	Tmax
● 파일 기준일자	%m%d
● 영업일 달력	CalID 🔍 검색
● 파일 기준일자 변동분	-1
● 업무코드	
● VAN ID	
● 후처리 ID	

파일전송 스타일

항목	설명
소스 파일 경로	소스 파일의 경로를 입력한다.
소스 파일 백업 경로	소스 파일을 백업할 경로를 입력한다.
로컬 파일 경로	로컬 파일의 경로를 입력한다.
로컬 파일 백업 경로	로컬 파일을 백업할 경로를 입력한다.
타겟 파일 경로	타겟 파일의 경로를 입력한다.
소스 파일명	소스 파일명을 입력한다.
로컬 파일명	로컬 파일명을 입력한다.
타겟 파일명	타겟 파일명을 입력한다.

◦ 한국 신용 정보원(KCREDIT) 스타일

한국 신용 정보원에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• 기관코드	CODE
• 기관명	Tmax
• 사용자 ID	USER_ID
• 사용자 PW	PSWD
• 헤더크기	42
• 데이터 패킷크기	4054
• 결번검증건수	5
• 레코드 크기	100
• 레코드 갯수	100
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID Q 검색
• 파일 기준일자 변동분	-1
• 중복수신 여부	Yes
• 업무코드	
• VAN ID	
• 후처리 ID	

한국 신용 정보원 스타일

◦ 한국 예탁 결제원(KSD) 스타일

한국 예탁 결제원에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

한국 예탁 결제원에서는 플로우의 "TCP MESSAGE SPLIT ACTIVITY"를 사용하여 데이터를 분할해서 수신해야 한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• 기관코드	CODE
• 기관명	Tmax
• 헤더크기	42
• 데이터 패킷크기	4054
• 레코드 크기	100
• 레코드 갯수	2
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID 🔍 검색
• 파일 기준일자 변동분	-1
• 중복수신 여부	Yes

한국 예탁 결제원

◦ 한국 정책 금융공사(KOTA) 스타일

한국 정책 금융공사에서 입력하는 일부 항목에 대한 설명은 [금융결제원 CMS 스타일](#)을 참조한다.

스타일 설정

• 파일코드	ABCD
• 설명	파일코드1
• 배치 파일 경로	REPOSITORY/ftp/folder
• 성공 재처리 파일 경로	REPOSITORY/ftp/folder2
• 실패 재처리 파일 경로	REPOSITORY/ftp/folder3
• 파일명	AAAAAA.txt
• 기관코드	CODE
• 기관명	Tmax
• 사용자 ID	USER_ID
• 사용자 PW	PSWD
• 암호화유형	사용 안함
• 헤더크기	42
• 데이터 패킷크기	4054
• 결번검증건수	5
• 레코드 크기	100
• 레코드 갯수	2
• 개행문자	설정 안함
• 파일 기준일자	%m%d
• 영업일 달력	CalID 🔍 검색
• 파일 기준일자 변동분	-1
• 중복수신 여부	Yes
• 업무코드	
• VAN ID	
• 후처리 ID	
• 송수신 FLAG	

한국 정책 금융공사

항목	설명
송수신 FLAG	송수신 FLAG를 입력한다.

코드 정보 설정

• 코드 정보

배치 플로우의 코드정보를 입력하여 입력된 코드 값 들을 응답 메시지로 매핑하여 사용할 수 있다.

코드 정보

개시 코드	
• 종별 코드	1001
• 거래 코드	2001
개별 업무 개시 코드	
• 종별 코드	1002
• 거래 코드	2002
종료 코드	
• 종별 코드	1003
• 거래 코드	2003
개별 업무 종료 코드	
• 종별 코드	1004
• 거래 코드	2004
결번 확인 코드	
• 종별 코드	1005
• 거래 코드	2005
헤더 코드	
• 종별 코드	1006
• 거래 코드	2006
데이터 코드	
• 종별 코드	1007
• 거래 코드	2007
트레일러 코드	
• 종별 코드	1008
• 거래 코드	2008
테스트 코드	
• 종별 코드	1009
• 거래 코드	2009
이어받기 코드	
• 종별 코드	1010
• 거래 코드	2010
파일 계속 코드	
• 종별 코드	1011
• 거래 코드	2011

매핑 설정

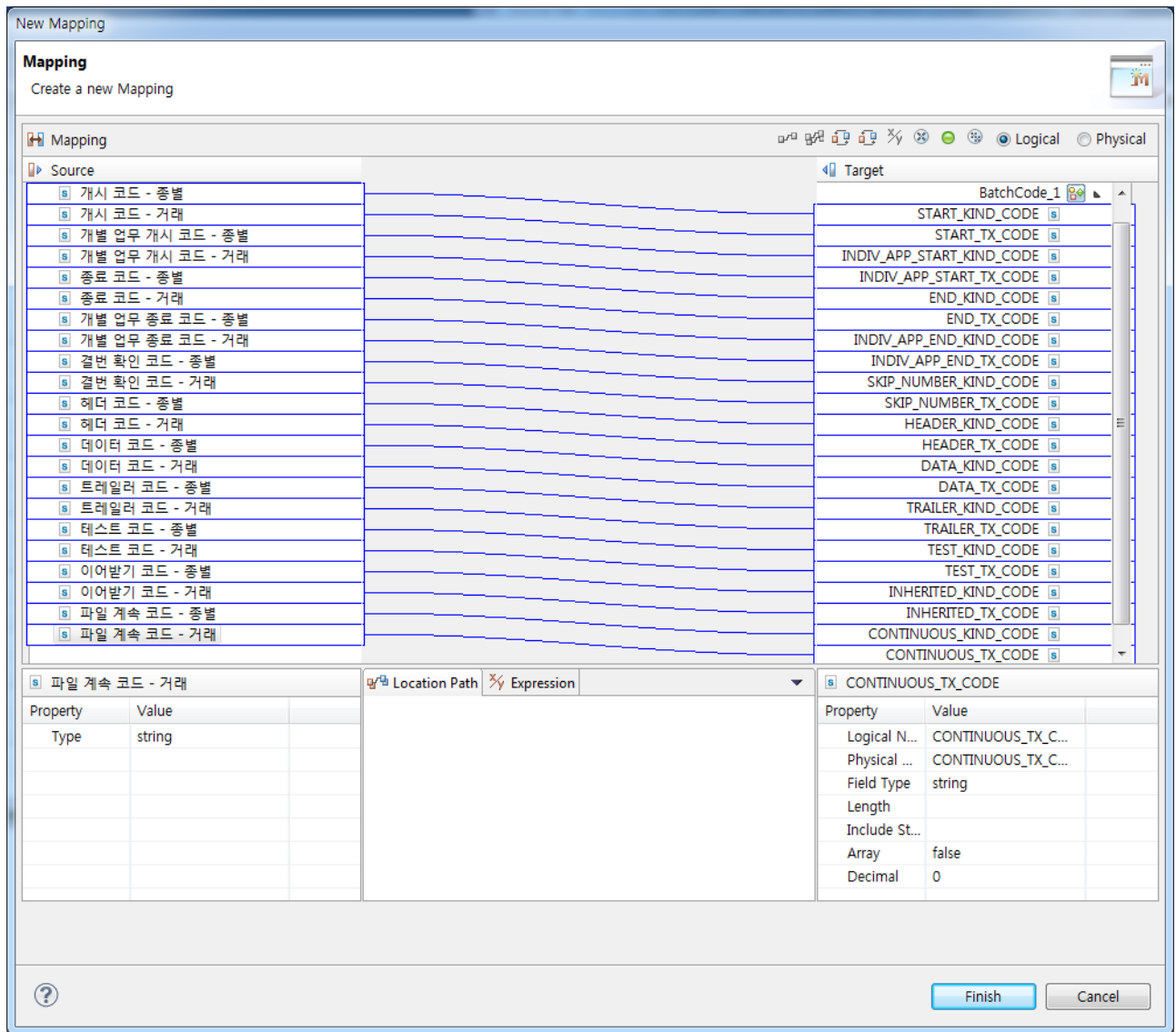
OutputMapping

코드 정보

항목	설정
종별코드	종별 코드를 입력한다.
거래 코드	거래 코드를 입력한다.
OutputMapping	코드 정보에 입력된 각 코드들을 룰에 설정된 응답 메시지에 매핑한다.

• OutputMapping

OutputMapping을 이용하여 입력한 코드들을 응답 메시지로 매핑할 수 있다.



OutputMapping

항목	설정
Source	코드 정보에서 입력한 필드들이 표시된다.
Target	응답 메시지가 표시된다.

5. 멀티바인딩 룰 에디터

본 절에서는 멀티바인딩 룰을 설정하는 에디터의 사용법에 대해서 설명한다.

5.1. 멀티바인딩 룰

멀티바인딩은 동적으로 서비스 내부적인 라우팅을 담당하는 것을 말하는 것으로, 라우팅을 담당하는 부분을 멀티바인딩 라우터라 말하며 서로 다른 서비스를 하나의 서비스로 묶어 내부의 라우팅을 실제로 담당시킬 수 있는 멀티바인딩 룰로 구성된다.

멀티바인딩 룰은 여러 개의 서비스를 하나로 묶어 조건에 따라 실행 도중 조건에 따라 플로우의 수행 순서를 분기하거나 여러 개의 서비스를 한꺼번에 호출하는 멀티캐스팅 기능을 제공한다.

멀티바인딩은 서비스의 메시지 이벤트(Receive), 어댑터의 아웃바운드 룰, 멀티바인딩 룰, 내부 서비스등을 모두 그룹핑하는 것이 가능하며 스스로도 하나의 서비스로 간주되는 '룰'이므로 어댑터의 파싱 룰에 의해 플로우 내부에서 즉시 멀티바인딩 룰로 호출될 수도 있다.

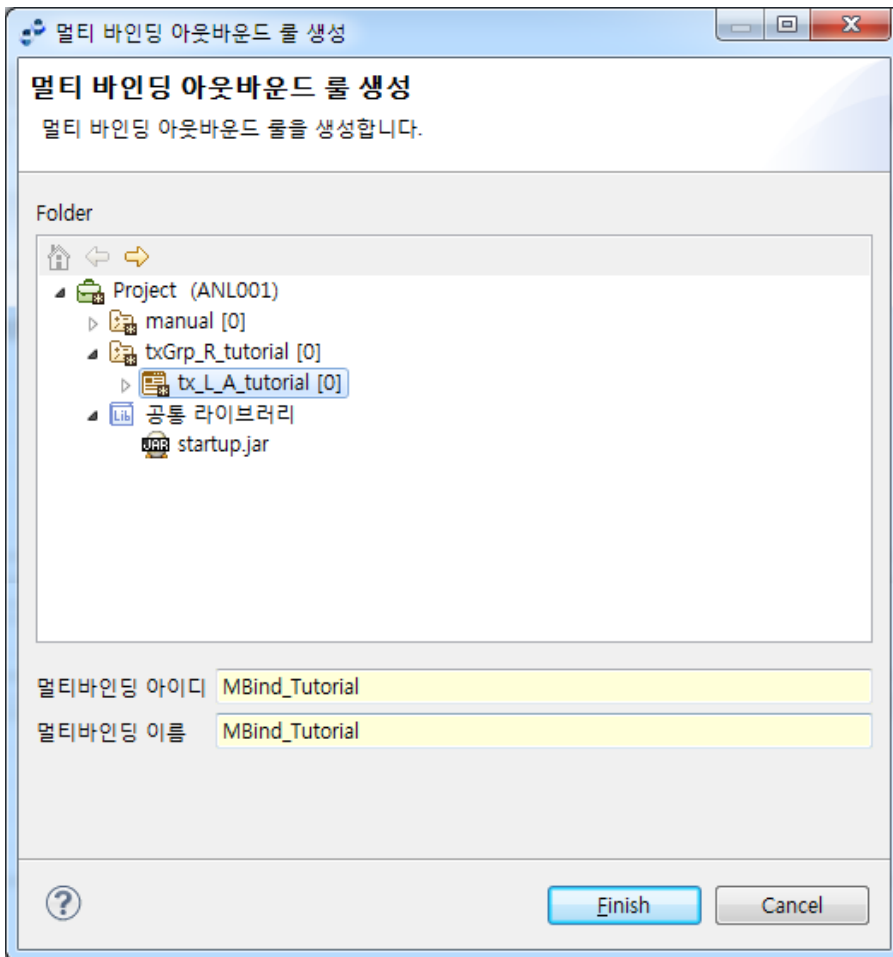
여러 서비스로 분기하는 것은 서비스 플로우 액티비티의 "게이트 액티비티"에서 사용하는 **Split** 개념과 유사하며 여러 액티비티를 한 번에 호출하는 멀티캐스팅은 서비스 플로우와 메시지 이벤트들 간의 코릴레이션을 수행한다고 보면 된다.



1. 멀티바인딩 룰에 대한 더욱 자세한 설명은 "AnyLink 런타임 엔진 서버 안내서"를 참고한다.
2. 플로우 에디터에서 멀티바인딩을 사용하기 위해서는 반드시 먼저 다음과 같이 멀티바인딩 룰을 생성한 후 플로우 에디터에서 관련 내용을 입력해야만 한다. 플로우 에디터를 사용하는 방법은 [서비스 플로우 에디터](#)를 참고한다.

5.2. 멀티바인딩 룰 생성

멀티바인딩 룰 생성은 왼쪽 프로젝트 네비게이터의 컨텍스트 메뉴에서 **[새로만들기] > [멀티바인딩]**을 선택한다. **멀티바인딩 생성 화면**에서 각 항목을 설정하고 **[Finish]** 버튼을 클릭한다.



멀티바인딩 룰 생성 화면

항목	설명
멀티바인딩 아이디	멀티바인딩의 아이디를 입력한다. 영어, 숫자, 특수문자(_)만 입력이 가능하며 첫 글자는 영어로 입력한다.
멀티바인딩 이름	멀티바인딩의 이름을 입력한다. 한글, 영어, 숫자, 특수문자 입력이 가능하다. 멀티바인딩 이름은 XML Naming Convention을 따른다.

5.3. 멀티바인딩 설정

멀티바인딩 에디터에서 멀티바인딩 룰에 필요한 정보를 작성한다.

• 기본 정보

멀티 바인딩 기본정보	
• 멀티 바인딩 ID	MBind_Tutorial
• 멀티 바인딩 이름	MBind_Tutorial

멀티바인딩 - 기본 정보

항목	설명
멀티바인딩 ID	리소스 아이디로 작성한 아이디이다. 수정은 불가능하다.
멀티바인딩 이름	멀티바인딩의 이름을 입력한다. 멀티바인딩의 이름은 중복을 허용하지 않으므로 다른 멀티바인딩의 이름과 겹치지 않도록 유의하도록 한다.
설명	멀티바인딩에 대한 설명을 기술한다. 각 멀티바인딩을 구별하기 쉽게 도와주므로 보다 상세하게 설정할 것을 권장한다.

• 요청/응답/업무 오류 메시지 설정

[추가] 버튼을 클릭하여 [메시지 선택 다이얼로그](#)에서 메시지를 추가하거나 [제거] 버튼을 클릭하여 추가한 메시지를 삭제할 수 있다.

요청 메시지

이름	메시지 ID	타입 ID
Req_Header_R...	Req_Header_R...	Req_Header_R_Tut...

가져오기
+ 추가
- 삭제

멀티바인딩 - 요청 메시지

항목	설명
요청 메시지	멀티바인딩 룰에 입력으로 들어오는 메시지의 형식을 지정한다.
응답 메시지	멀티바인딩 결과를 메시지로 따로 출력할 때 성공적으로 작업을 마친 경우 보낼 메시지의 형식을 지정한다.
업무 오류 응답 메시지	멀티바인딩 결과를 메시지로 따로 출력할 때 작업 도중 오류가 발생한 경우 보낼 메시지의 형식을 지정한다.

• 바인딩 설정

바인딩에 요구되는 각종 요소들을 설명한다. 자세한 내용은 [매핑 설정](#)을 참고한다.

바인딩 설정

선택 옵션

TIME_RANGE

선택 옵션 추가 정보

값	이름	ID	타입	요청 매핑	응답 매핑	장애응답 매핑
01:25-23:20	Out8_MB_TCP_TC_0003_SVC1	Out8_MB_TCP_TC_0003_SVC1	OUTBOUND_RULE	MB_SVC_TC_0003_01RuleInputMap_0.map	MB_SVC_TC_0003_01RuleOutputMap_0.map	
23:35-23:38	Out8_MB_TCP_TC_0003_SVC2	Out8_MB_TCP_TC_0003_SVC2	OUTBOUND_RULE	MB_SVC_TC_0003_01RuleInputMap_1.map	MB_SVC_TC_0003_01RuleOutputMap_1.map	

+

추가

-

삭제

멀티바인딩 - 바인딩 설정

항목	설명
선택 옵션	<p>멀티바인딩에 사용할 라우팅 메소드 방법을 설정한다.</p> <ul style="list-style-type: none"> Value : 선택 옵션 추가 정보에서 설정한 표현식의 결과값에 따라 라우팅을 수행한다. RoundRobin : 각 서비스를 순차적으로 라우팅한다. WeightBased : RoundRobin과 같으나 서비스의 가중치를 고려해 라우팅한다. Multicast : 등록된 모든 서비스/메시지에 메시지를 전달한다. One-way 방식의 서비스에서만 사용할 수 있다. TimeRange : 지정한 시간대에 서비스를 호출시킨다. Flow Correlation : 서비스 플로우의 코릴레이션. 가장 먼저 값이 일치하는 서비스 항목으로 라우팅한다. 서비스 항목은 코릴레이션을 지원하는 서비스 플로우 메시지 이벤트여야 한다. Handler : 사용자가 정의한 핸들러를 이용하여 라우팅 방법을 결정한다. 선택 옵션 추가 정보에 사용할 핸들러의 이름을 적어야 한다. <p>선택 옵션에서 타입을 변경하면 매핑 목록에 입력한 내용이 모두 삭제되므로 주의한다.</p>
선택 옵션 추가 정보	<p>'선택 옵션' 항목에서 선택한 라우팅 메소드 방법에 추가적으로 요구되는 항목을 적는다. Value, Handler에서만 사용되며 나머지의 경우는 적지 않는다.</p> <ul style="list-style-type: none"> Value : 값을 판단할 표현식의 이름을 적는다. 여기서 정의한 표현식에 따라 어느 서비스로 라우팅할지 결정된다. Handler : 라우팅할 대상을 결정하는 핸들러의 클래스의 이름을 적는다.
매핑 목록	<p>라우팅의 조건에 따라 라우팅할 서비스를 선택하고, 멀티바인딩의 메시지들과 서비스의 메시지들을 서로 매핑한다.</p>

5.4. 매핑 설정

본 절에서는 [바인딩 설정] > [매핑 목록]에서 메시지들을 매핑하는 방법을 설명한다.

5.4.1. 매핑 추가/삭제

매핑은 하단의 **[추가]**와 **[삭제]** 버튼을 클릭하면 설정할 수 있다. **[추가]** 버튼을 클릭하면 **매핑 목록**에 다음 그림과 같이 공백인 줄이 생성되며 각 필드를 클릭하여 데이터를 설정할 수 있다. 생성된 매핑의 제거는 **[삭제]** 버튼을 클릭한다.

[illegible]

+ 추가
- 삭제

매핑 추가/삭제 화면

다음은 매핑 값을 설정하는 항목에 대한 설명이다.

항목	설명
값	[바인딩 설정] > [선택 옵션] 에서 선택한 라우팅의 조건을 적는다. 예를 들어 Value로 설정된 경우 입력한 값과 일치한 값이 입력으로 들어올 때 설정한 서비스로 라우팅된다.
이름	라우팅할 서비스의 이름을 표기한다. 사용자가 편집할 수 없고 '아이디' 항목으로 서비스가 설정된 경우 해당 서비스의 이름을 표기한다.
ID	라우팅할 서비스의 ID를 표기한다. '아이디' 항목을 클릭하면 [찾기] 버튼이 나타난다. 해당 버튼을 클릭하면 리소스 검색 화면에서 원하는 서비스를 찾아 등록할 수 있다.
타입	라우팅할 서비스의 타입을 표기한다. 사용자가 편집할 수 없고 '아이디' 항목으로 서비스가 설정된 경우 해당 서비스의 타입을 표기한다.
요청 매핑	멀티바인딩 룰로 입력으로 들어오는 메시지와 라우팅되는 서비스의 입력 메시지를 매핑하는 방법을 설정한다. 컬럼을 클릭하면 [찾기] 버튼과 [추가] 버튼이 나타난다. <ul style="list-style-type: none">◦ [찾기] 버튼 : 이미 설정된 매핑 파일을 리소스 검색 화면을 통해 찾아 매핑하는 방법을 설정한다. 이미 매핑이 정의된 경우 유용하다.◦ [추가] 버튼 : 매핑 화면을 통해 메시지들을 서로 매핑한다.

항목	설명
응답 매핑	<p>라우팅된 서비스의 정상 응답 메시지를 멀티바인딩 정상 응답 메시지와 매핑하는 방법을 설정한다.</p> <p>컬럼을 클릭하면 [찾기] 버튼과 [추가] 버튼이 나타난다. 기능 설명은 '요청 매핑' 항목의 설명을 참고한다.</p>
장애응답 매핑	<p>라우팅된 서비스의 업무오류 응답 메시지를 멀티바인딩 업무오류 응답 메시지와 매핑하는 방법을 설정한다.</p> <p>컬럼을 클릭하면 [찾기] 버튼과 [추가] 버튼이 나타난다. 기능 설명은 '요청 매핑' 항목의 설명을 참고한다.</p>

6. 서비스 플로우 에디터

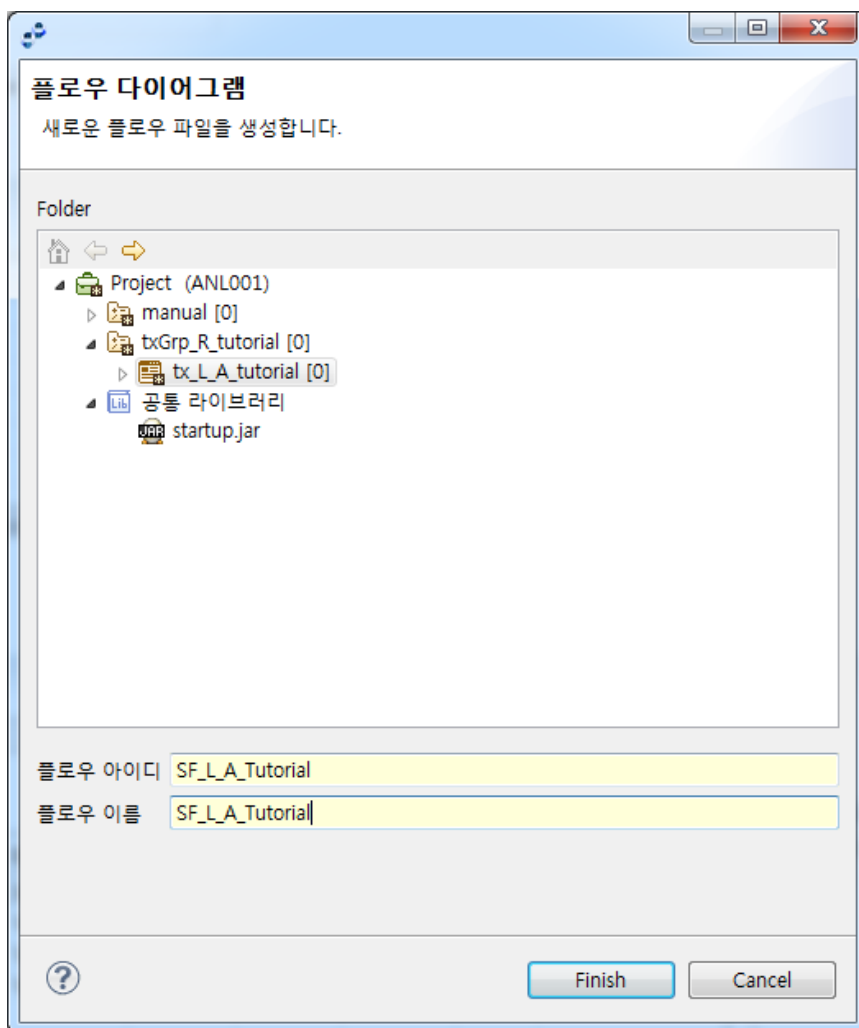
본 장에서는 AnyLink 스튜디오의 서비스 플로우 에디터의 기능과 사용법에 대해 설명한다.

6.1. 개요

서비스 플로우는 서비스의 수행 절차를 그래프로 표현한 것이다. 서비스 플로우 다이어그램(SFDL: Service Flow Definition Language) 파일을 생성하기 전에 AnyLink 프로젝트가 생성되어 있어야 한다. 서비스 플로우의 수신(Receive) 메시지 이벤트들이 AnyLink의 내부 서비스로 등록된다. 대부분 AnyLink 내부 서비스는 서비스 플로우의 메시지 이벤트이며, 플로우에 정의된 흐름을 따라 진행 후 플로우의 서비스 액티비티를 호출하게 된다.

6.2. 서비스 플로우 에디터 생성

서비스 플로우 생성은 왼쪽 프로젝트 네비게이터의 컨텍스트 메뉴에서 **[새로만들기] > [플로우]**를 선택한다. **플로우 다이어그램**에서 각 항목을 설정하고 **[Finish]** 버튼을 클릭한다.

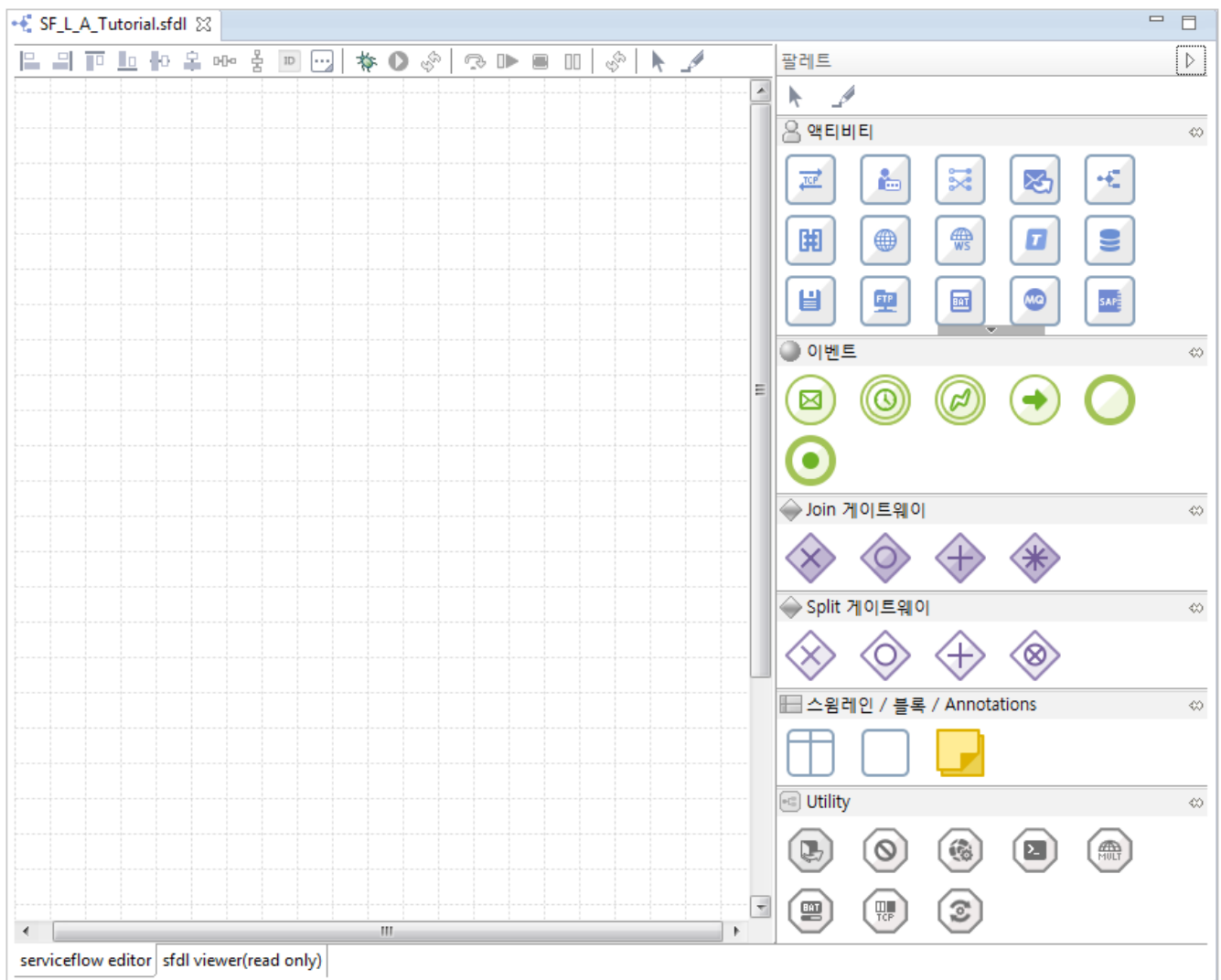


플로우 다이어그램

항목	설명
플로우 아이디	서비스 플로우 리소스 아이디를 입력한다. 영어, 숫자, 특수문자(_)만 사용이 가능하며 첫 글자는 영어로 입력한다.
플로우 이름	사용할 플로우명을 입력한다. 한글, 영어, 숫자, 특수문자 입력이 가능하다. 플로우 이름은 XML Naming Convention을 따른다.

6.3. 서비스 플로우 다이어그램

서비스 플로우를 생성하면 기본 화면이 나오며 오른쪽에 다이어그램을 추가할 수 있는 팔레트와 편집창 상단 툴바가 있다.



서비스 플로우 화면

• [툴바]

툴바는 플로우 다이어그램의 정렬, 화면 표시방법 및 디버거 동작을 위한 버튼으로 구성된다.



서비스 플로우 화면 - 톨바

버튼	설명
	다이어그램을 왼쪽으로 정렬한다.
	다이어그램을 오른쪽으로 정렬한다.
	다이어그램을 위쪽으로 정렬한다.
	다이어그램을 아래쪽으로 정렬한다.
	다이어그램을 중앙(상하)으로 정렬한다.
	다이어그램을 중앙(좌우)으로 정렬한다.
	다이어그램들의 수평 거리를 일정하게 정렬한다.
	다이어그램을 수직 거리를 일정하게 정렬한다.
	다이어그램을 ID/Name 표시를 변경한다.
	Transition Condition 표시 유무를 변경한다.
	디버거 모드를 실행시킨다. 서버 선택 다이얼로그가 나타난다.
	디버깅을 위한 메시지 설정 다이얼로그가 나타난다.
	현재 디버거 상태를 조회한다.
	디버거를 다음 액티비티로 이동시킨다.
	디버거를 다음 Break Point로 이동시킨다.
	디버거를 플로우 끝까지 이동시킨다.
	디버거가 Running 상태인 경우 다음 액티비티에서 정지시킨다.
	디버거 상태를 나타내는 아이콘이다. (버튼 아님)
	마우스 포인터를 'Selection' 상태로 변경한다.
	마우스 포인터를 'Transition' 상태로 변경한다.



• [팔레트]

팔레트는 서비스 플로우에서 제공하는 서비스 플로우 다이어그램이다.

서비스 플로우는 프로세스(Process)라고도 부르며, 액티비티, 이벤트와 트랜지션으로 구성된다. 원하는 객체를 클릭하여 에디터 영역에 올려 사용한다.











서비스 플로우 화면 - 팔레트





구분	설명
Selection, Transition	<ul style="list-style-type: none"> •  (Select) : 트랜지션 설정을 종료한다. •  (Transition) : 트랜지션은 액티비티 또는 이벤트들 간의 흐름의 순서를 나타내는 화살표이다. 에디터의 각 객체를 연결할 수 있다. 각 단위 액티비티 간의 흐름을 표시하는 것으로서, 실제로 업무 프로세스의 흐름을 액티비티, 이벤트, 게이트웨이 간 연결하여 표시한다. 트랜지션의 또 하나의 기능은 게이트웨이 이후에 조건에 따라 업무 흐름을 결정할 때 조건을 설정하는 것이다.
액티비티(Activity)	<p>작업(Task)이라고도 부르며, 서비스 플로우에 해당 위치로 흐름이 도달하게 되면 실행해야 할 어떤 일들을 표현한다.</p> <p>플로우에 그려진 서비스 액티비티는 대부분 어댑터의 아웃바운드 룰 서비스를 의미한다. 자세한 설명은 액티비티를 참고한다.</p>

구분	설명
이벤트(Event)	<p>사건을 나타내는 특별한 액티비티로 메시지, 에러, 타임아웃 등의 사건을 나타낸다.</p> <p>이벤트는 전후 트랜지션 유무에 따라 들어오는 트랜지션이 없는 시작 이벤트(Start Event), 들어오는 트랜지션과 나가는 트랜지션이 모두 있는 중간 이벤트(Intermediate Event), 나가는 트랜지션이 없는 끝 이벤트(End Event)로 구분할 수 있다. 자세한 설명은 이벤트를 참고한다.</p>
게이트웨이(Gateway)	<p>흐름을 제어하는데 사용되는 특별한 액티비티 형태라고 생각할 수 있다.</p> <p>크게 하나의 트랜지션 흐름을 여러 개의 트랜지션 흐름으로 분리시켜주는 스플릿 게이트웨이(Split Gateway)와 그 반대로 여러 개의 트랜지션 흐름을 하나의 트랜지션 흐름을 합쳐주는 조인 게이트웨이(Join Gateway)가 있다. 자세한 설명은 게이트웨이를 참고한다.</p>
스왐레인 / 블록 / Annotations	<ul style="list-style-type: none"> 스왐레인 : 서비스 플로우 상의 Task를 구분해주는 선을 나타낸다. 블록 : 특정 구역을 묶어서 표현하기 위해 사용되는 액티비티 묶음이다. Annotations : 플로우의 메모 기능을 제공하기 위한 객체이다. <p>자세한 설명은 스왐레인 / 블록 / ANNOTATIONS를 참고한다.</p>
Utility	서비스 플로우에서 사용되는 Utility이다. 자세한 설명은 Utility 를 참고한다.

6.4. 액티비티

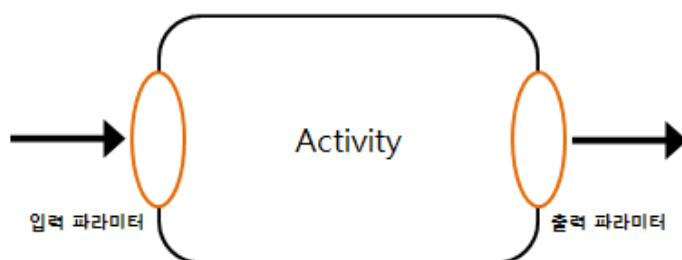
다음은 액티비티의 종류에 대한 설명이다.

0메뉴	설명
 (TCP 액티비티)	TCP 아웃바운드 룰 서비스를 호출할 수 있는 액티비티이다.
 (유저 클래스 액티비티)	유저 클래스를 사용할 수 있는 액티비티이다.
 (매핑 액티비티)	Source와 Target 변수를 설정하여 매핑해 주는 액티비티이다.
 (응답 메시지 액티비티)	서비스의 응답 메시지로 사용할 수 있는 액티비티이다. 이벤트 메시지의 요청 메시지로부터 응답 메시지를 보내준다. 정상 응답과 오류 응답 2개의 형태로 구현할 수 있다.
 (서브 플로우 액티비티)	다른 서비스 플로우 프로세스를 호출하는 액티비티이다.
 (멀티바인딩 액티비티)	멀티바인딩 서비스를 호출할 수 있는 액티비티이다.
 (HTTP 액티비티)	HTTP 아웃바운드 룰 서비스를 호출할 수 있는 액티비티이다.
 (웹서비스 액티비티)	웹 서비스 형태로 만들어진 서비스를 import하여 사용할 수 있는 액티비티이다.

O메뉴	설명
 (TMAX 액티비티)	미들웨어인 Tmax와 관련된 서비스를 호출할 수 있는 액티비티이다.
 (DB 액티비티)	DB 어댑터 서비스를 호출할 수 있는 액티비티이다.
 (파일 액티비티)	File 어댑터 서비스를 호출할 수 있는 액티비티이다.
 (FTP 액티비티)	FTP 어댑터 서비스를 호출할 수 있는 액티비티이다.
 (배치 액티비티)	배치 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (MQ 액티비티)	MQ 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (SAP 액티비티)	SAP 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (SMTP 액티비티)	SMTP 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (WebDav 액티비티)	WebDav 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (Tuxedo 액티비티)	Tuxedo 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (JMS 액티비티)	JMS 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (ISO8583 액티비티)	ISO8583 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (UDP 액티비티)	UDP 아웃바운드 룰을 호출할 수 있는 액티비티이다.
 (ProObject 액티비티)	ProObject 아웃바운드 룰을 호출할 수 있는 액티비티이다.

액티비티는 입력, 출력에 해당하는 파라미터를 가진다.

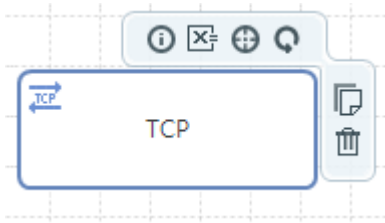
입력 파라미터는 해당 액티비티가 읽기 위해 사용하는 변수이고, 출력 파라미터는 해당 액티비티가 쓰기 위해 사용하는 변수이다. 두 변수 모두 프로세스나 액티비티를 둘러싼 블록 액티비티에 선언된다.



액티비티의 입력과 출력

6.4.1. 액티비티 설정

액티비티를 에디터 화면으로 갖고 온 뒤 더블클릭 또는 마우스를 객체 위로 올리면 나타나는 **Activity Preference 화면**에서 액티비티 설정을 할 수 있다.

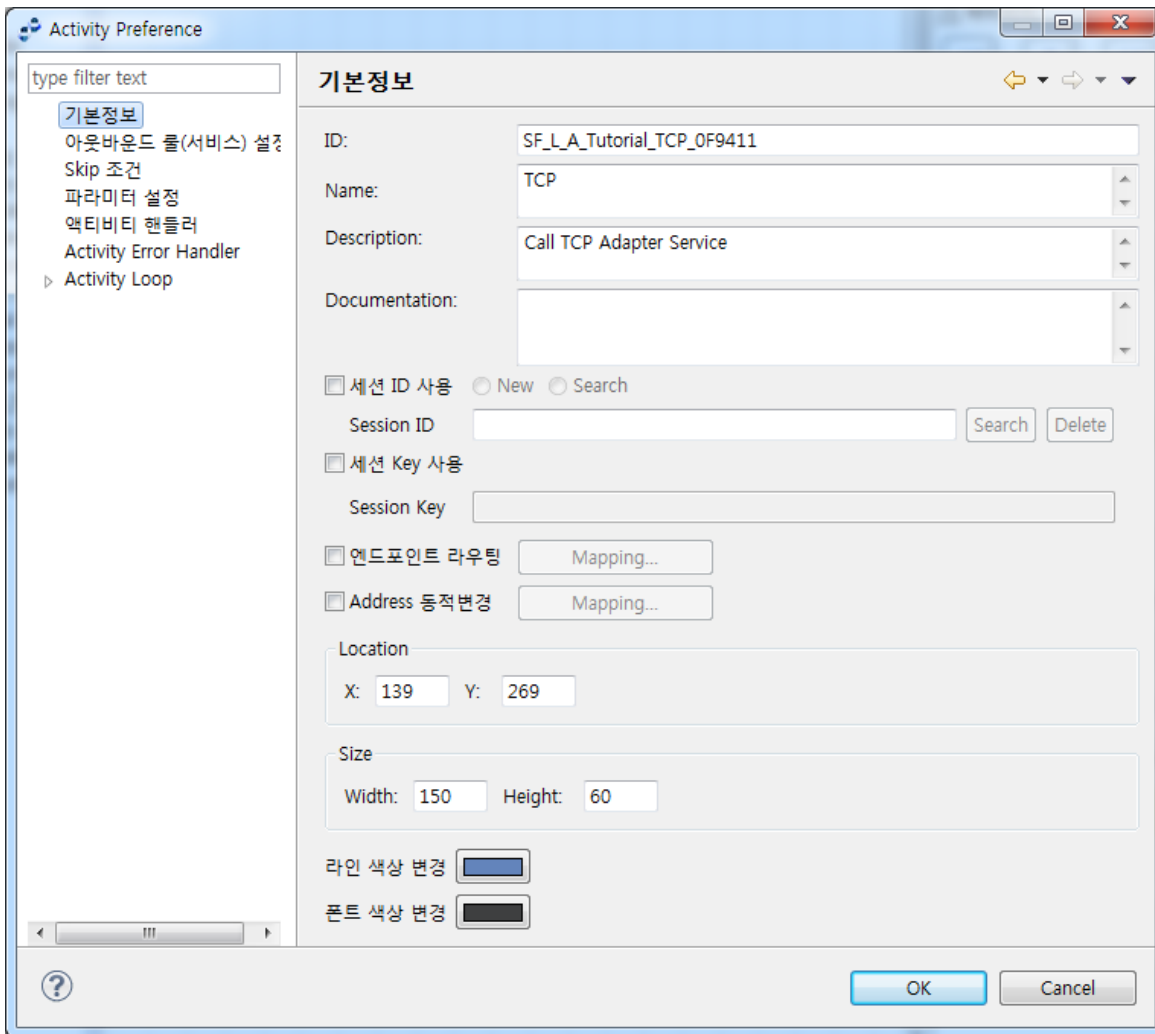


Activity Preference 화면

메뉴	설명	메뉴	설명
i	일반정보	Q	Loop 설정
x	파라미터 설정	Copy icon	복사
+	핸들러 설정	Trash icon	삭제

기본 정보

다음은 활성화된 **Activity Preference 화면**에 대한 설명이다.



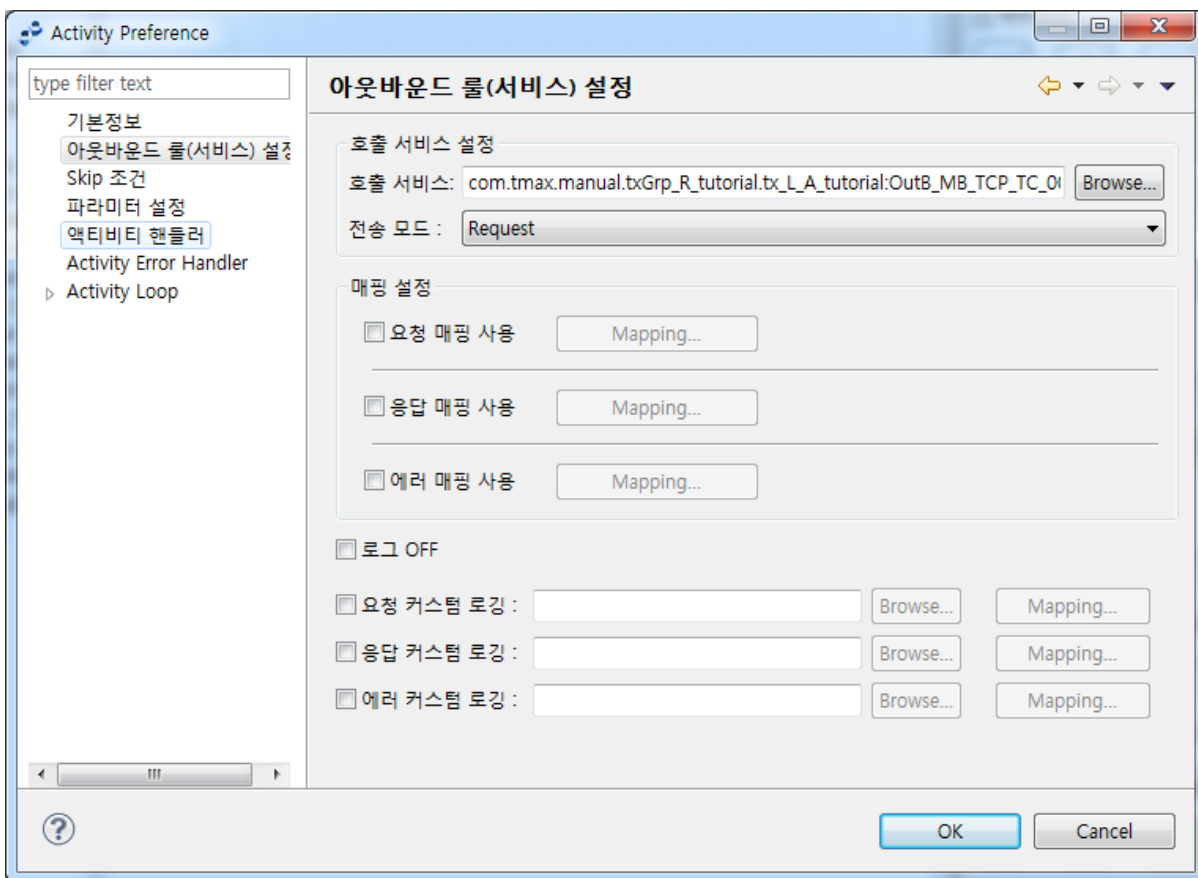
Activity Preference 화면 - 기본 정보

항목	설명
ID	해당 객체의 아이디를 입력할 수 있는 항목이다.
Name	해당 객체 이름을 알려주는 항목이다.
Description	해당 액티비티의 역할을 간략하게 나타낸다.
Documentation	해당 액티비티에 대한 설명으로 사용자의 편의를 위한 항목이다.
Session ID	입력한 Session ID를 가지는 다른 액티비티, 이벤트들과 동일한 세션으로 연결하기 위한 항목이다.
Session Key	입력한 세션 키를 가지는 세션으로 연결하기 위한 항목이다. TCP 액티비티에서만 설정 가능하다.
엔드포인트 라우팅	<p>매핑에서 설정된 VALUE를 가진 엔드포인트나 엔드포인트 그룹에 라우팅을 하기 위한 항목이다.</p> <p>아웃바운드를 설정한 엔드포인트 그룹이 WebAdmin의 [구성관리] > [어댑터]의 엔드포인트 그룹에서 매핑으로 설정되어 있어야 한다. 하위 엔드포인트/엔드포인트 그룹의 VALUE 설정시 해당 엔드포인트로 요청메시지가 전달된다.</p>
Address 동적변경	TCP 클라이언트인 경우 요청할 대상의 IP, Port를 동적으로 변경하기 위한 항목이다.
URL 동적변경	HTTP 아웃바운드의 경우 요청할 대상의 IP, Port를 동적으로 변경하기 위한 항목이다.

항목	설명
Location	객체의 위치를 나타내는 x, y 좌표값이다. 좌표 설정을 통해 액티비티의 위치를 변경할 수 있지만, 에디터에서 마우스 드래그를 통해 옮기는 것을 추천한다.
Width	객체의 위치를 나타내는 값으로, Location과 마찬가지로 에디터에서 드래그로 크기를 조절 가능하다.
라인 색상 변경	객체의 외곽 라인 색상을 변경시킬 수 있는 항목이다.
폰트 색상 변경	객체의 글자 폰트 색상을 변경시킬 수 있는 항목이다.

아웃바운드 룰(서비스) 설정

다음은 **Activity Preference** 화면의 [아웃바운드 룰(서비스) 설정]에 대한 설명이다.



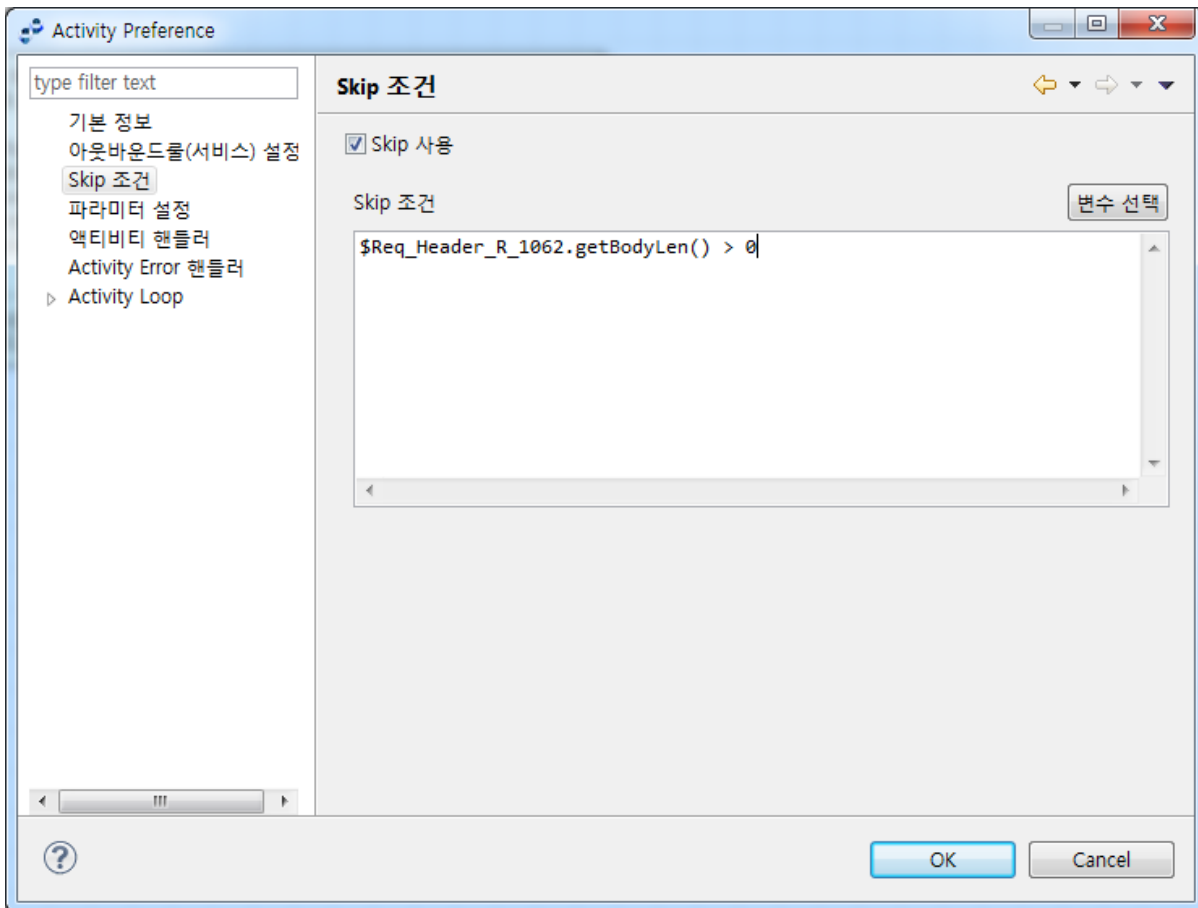
Activity Preference 화면 - 아웃바운드 룰 설정

항목	설명
호출서비스	사전에 생성한 아웃바운드 룰 서비스를 검색하여 적용시킬 수 있는 항목이다. 단, 액티비티에 맞는 아웃바운드 룰을 적용시켜야 한다. [Browse...] 버튼을 클릭하면 서비스를 검색할 수 있는 다이얼로그가 생성된다.

항목	설명
전송모드	<ul style="list-style-type: none"> ◦ Request : 가장 일반적인 메시징 형태로 요청에 대한 응답을 기대하는 전송모드이다. ◦ One Way : 응답을 기다리지 않고 요청 메시지만 전달하는 전송모드이다. ◦ One Way with ack : 요청 메시지가 처리된 후 ACK 메시지를 기대하는 형태이다. 보통 딜리버리 채널에서 전달 보장을 처리한 후 ACK 메시지 또는 NAK 메시지를 답변으로 보내는 전송모드이다.
매핑 설정	<ul style="list-style-type: none"> ◦ 요청 매핑 사용 : 전 서비스 플로우의 INPUT 파라미터와 자신의 요청 메시지에 관한 매핑을 정의한다. ◦ 응답 매핑 사용 : 자신의 응답 메시지와 OUPUT 파라미터에 관한 매핑을 정의한다. ◦ 에러 매핑 사용 : 에러 관련해서 설정한 메시지와 자신의 파라미터 변수와의 매핑을 정의한다. <p>매핑과 관련된 내용은 매핑 관련 다이얼로그를 참조한다.</p>
로그 OFF	해당 액티비티에 대한 로그를 OFF한다.
요청 커스텀 로깅	<p>요청 커스텀 로깅을 사용한다.</p> <p>[Browse...] 버튼을 클릭하면 커스텀 로그 아웃바운드 룰을 설정하고 [Mapping...] 버튼을 클릭해서 매핑을 설정한다.</p>
응답 커스텀 로깅	<p>응답 커스텀 로깅을 사용한다.</p> <p>[Browse...] 버튼을 클릭하면 커스텀 로그 아웃바운드 룰을 설정하고 [Mapping...] 버튼을 클릭해서 매핑을 설정한다.</p>
에러 커스텀 로깅	<p>에러 커스텀 로깅을 사용한다.</p> <p>[Browse...] 버튼을 클릭하면 커스텀 로그 아웃바운드 룰을 설정하고 [Mapping...] 버튼을 클릭해서 매핑을 설정한다.</p>

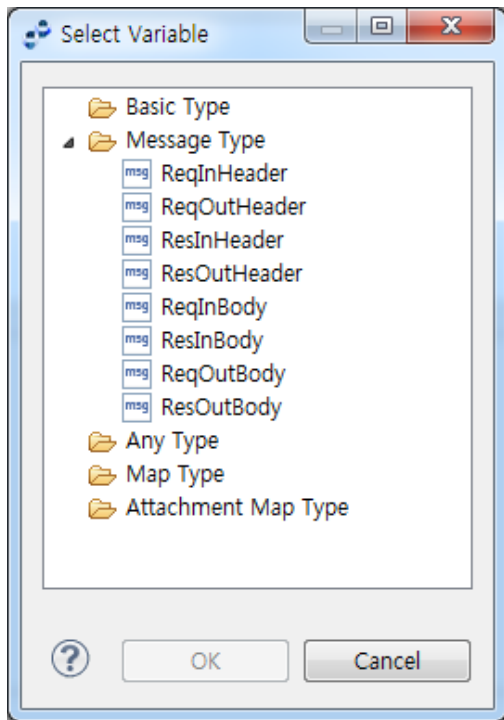
Skip 조건

다음은 **Activity Preference 화면의 Skip 조건**에 대한 설명이다.



Activity Preference 화면 - Skip 조건

항목	설명
Skip 사용	Skip 항목을 선택하면 프로세스 변수에 대한 Skip 조건을 사용할 수 있다.
Skip 조건	<p>선택한 변수에 대한 조건문을 작성해 주어야 한다.</p> <p>[변수 선택] 버튼을 클릭하면 프로세스 변수를 선택할 수 있는 Select Variable 화면 (Activity Preference 화면 - Select Variable 화면)이 나타난다.</p>

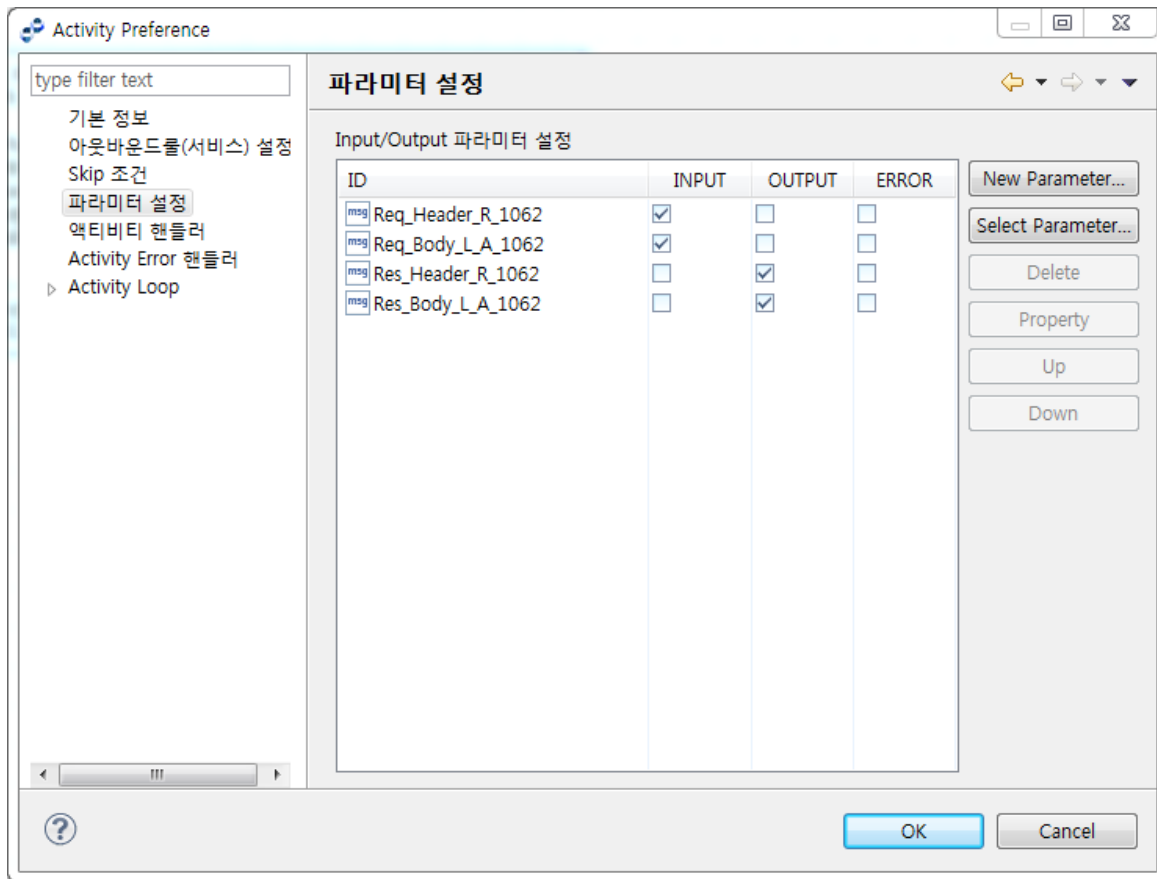


Activity Preference 화면 - Select Variable 화면

파라미터 설정

파라미터로 설정된 변수가 실제 서비스 호출에 이용된다. 매핑이 있다면 Input 파라미터는 Input 매핑의 소스가 되고, Output 파라미터는 Output 매핑에서 타겟이 된다.

다음은 **Activity Preference** 화면의 **[파라미터 설정]**에 대한 설명이다.



Activity Preference 화면 - 파라미터 설정

- Input/Output 파라미터 설정

항목	설명
ID	변수 파라미터 아이디이다.
INPUT	해당 변수 타입을 Input으로 설정한다.
OUTPUT	해당 변수 타입을 Output으로 설정한다.
ERROR	해당 변수 타입을 Error로 설정한다.

- 버튼

버튼	설명
[New Parameter]	새로운 파라미터를 생성할 수 있다. 버튼을 클릭하면 Add New Process Variable 화면 (Activity Preference 화면 - Add New Process Variable)이 나타난다.
[Select Parameter]	이미 생성되어 있는 파라미터를 선택할 수 있는 Select Variable 화면 (Activity Preference 화면 - Select Variable 화면)이 생성된다.
[Delete]	파라미터 화면에서 선택한 파라미터를 삭제한다.
[Property]	선택된 파라미터 정보를 보여준다.
[Up]	선택한 파라미터의 위치를 한 칸 높인다.
[Down]	선택한 파라미터의 위치를 한 칸 내린다.

[New Parameter] 버튼을 클릭하면 **Add New Process Variable** 화면이 나타난다.

The screenshot shows the 'Add New Process Variable' dialog box. It contains the following fields and options:

- Process Variable**
 - ID: Req_Body_L_A_Tutorial
 - Name: Req_Body_L_A_Tutorial
 - Description: (empty text area)
 - Initial Value: (empty text field)
 - Scope: instance (dropdown menu)
- Variable Type**
 - ☐ Basic Type
 - Variable Type: String (dropdown menu)
 - ☒ Message Type
 - Message ID: Req_Body_L_A_106 (text field) with a 'Browse...' button
 - ☐ Any Type
 - ☐ Map Type with a 'Key Setting...' button
 - ☐ Attachment Map Type with a 'Key Setting...' button
- ☐ Is Array Variable
- Buttons: ? (help), OK, Cancel

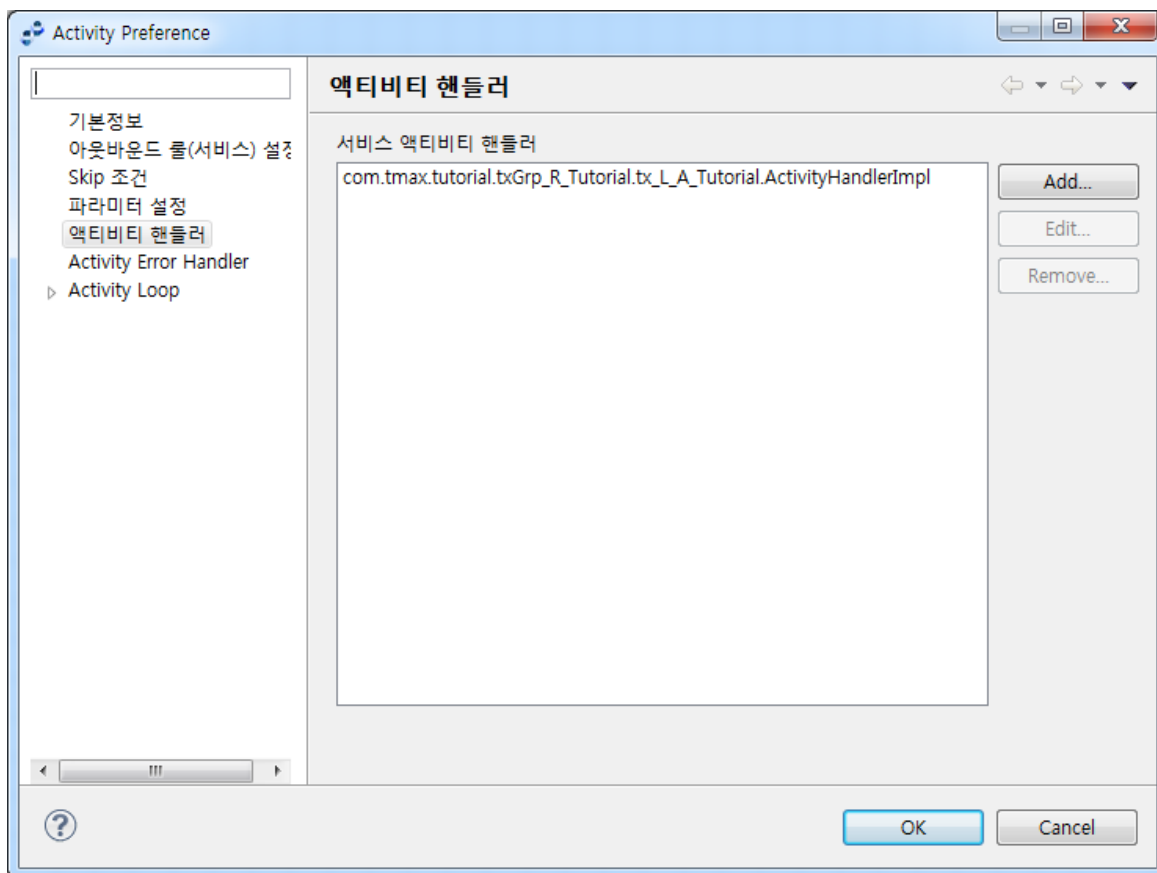
Activity Preference 화면 - Add New Process Variable

항목	설명
ID	생성할 파라미터의 아이디이다.
Name	생성할 파라미터의 이름이다.
Description	생성할 파라미터에 대한 설명으로 사용자의 편의를 위한 것이다.
Initial Value	생성할 파라미터의 타입이 ' Basic '일 경우, 변수에 대한 초기값을 설정할 수 있다.
Scope	인스턴스로 고정된 값이다.

항목	설명
Variable Type	<ul style="list-style-type: none"> • Basic Type : 변수의 기본 타입으로 String, Float, Integer, DateTime, Boolean Type을 지원한다. • Message Type : 메시지 타입으로, 자신의 프로젝트의 서비스 그룹 아래에 있는 DTO 파일들을 찾아서 선택하여 메시지 선택 다이얼로그에서 추가한다. • Any Type : Java Class 객체를 Process Variable로 사용하는 경우 해당 타입을 선택한다. • Map Type : Key Value, msg, ctx 정보를 접근하는 방법을 나타낸다. <p>[Key Setting] 버튼을 클릭해서 'Map Data Field Key' 값을 설정할 수 있다. 서비스 호출에 사용되는 header variable에는 Map Type만이 이용될수 있다. 형식은 <String, String>으로 설정한다.</p> <ul style="list-style-type: none"> • Attachment Map Type : 첨부 파일 등의 지정이 가능하며, 일반적으로 파일 Adapter Attachment 타입인 경우 사용한다.
Is Array Variable	배열을 사용할지 여부를 결정한다.

액티비티 핸들러

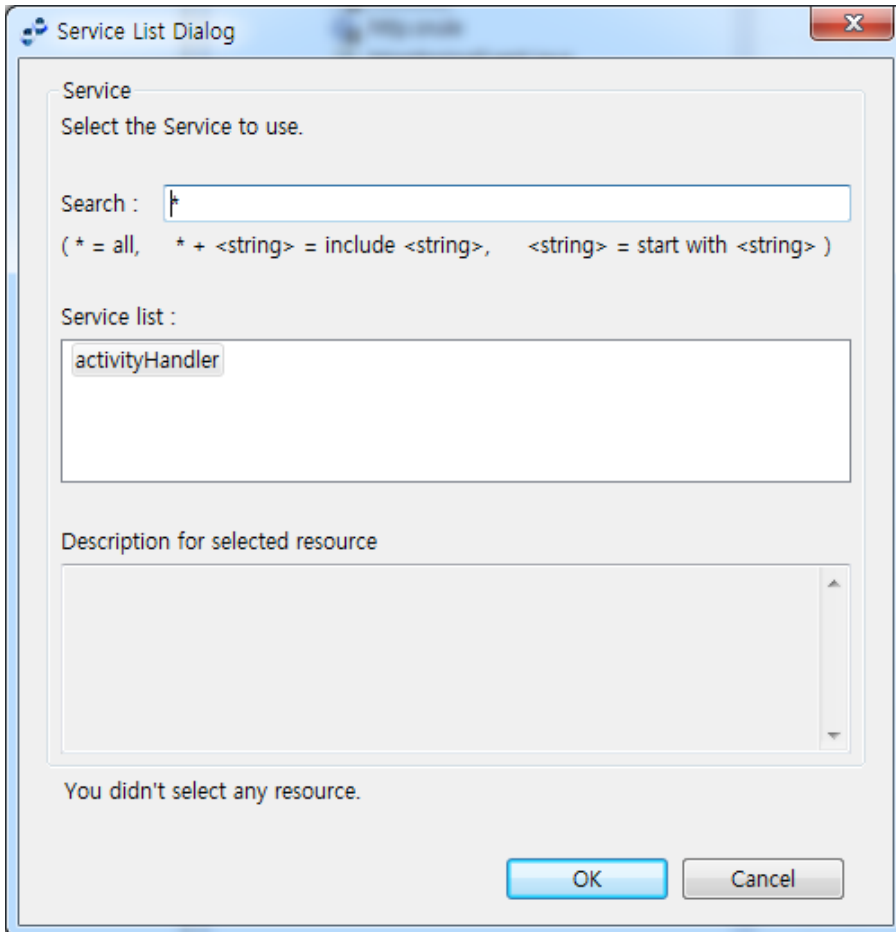
다음은 **Activity Preference** 화면의 [액티비티 핸들러]에 대한 설명이다.



Activity Preference 화면 - 액티비티 핸들러

버튼	설명
[Add]	사용자가 생성한 액티비티 핸들러를 설정할 수 있는 Service List Dialog 가 출력된다.
[Edit]	선택한 핸들러에 대한 수정 화면이 생성된다.
[Remove]	선택한 핸들러를 삭제한다.

다음은 사용자가 생성한 액티비티 핸들러를 설정할 수 있는 **Service List Dialog**에 대한 설명이다.

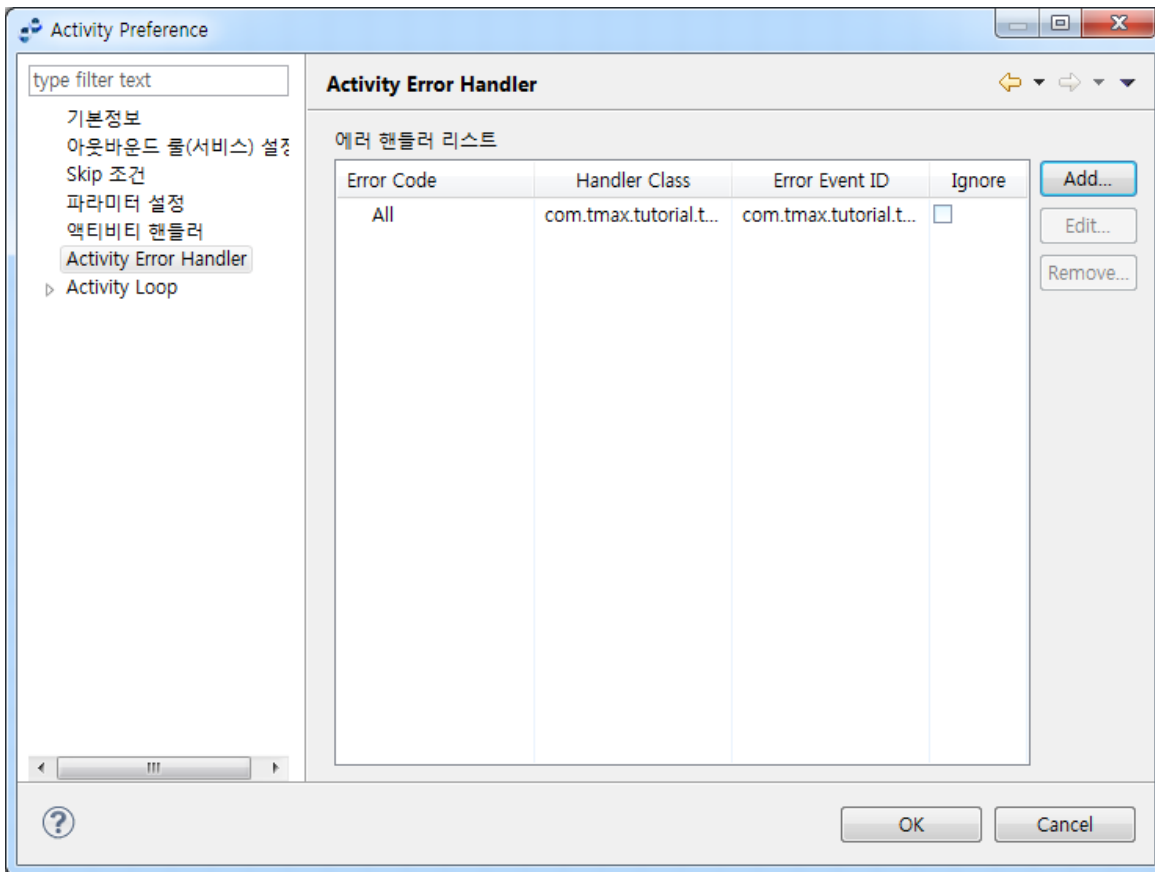


Service List Dialog

항목	설명
Search	액티비티 핸들러를 검색한다.
Service list	핸들러 목록이 나타난다.

Activity Error Handler

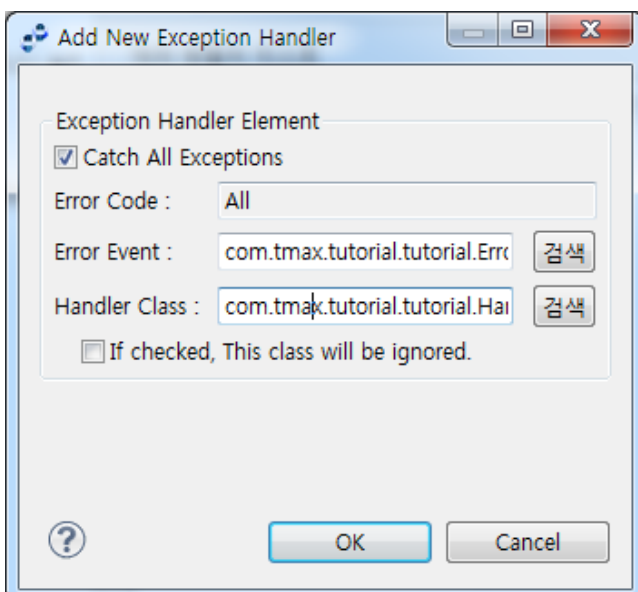
다음은 **Activity Preference 화면**의 **[Activity Error Handler]**에 대한 설명이다. 버튼의 기능은 **액티비티 핸들러 화면**([Activity Preference 화면 - 액티비티 핸들러](#))의 기능과 동일하다.



Activity Preference 화면 - Activity Error 핸들러

항목	설명
Error Code	핸들러가 처리할 에러 코드를 입력한다.
Handler Class	User Class를 직접 등록해서 처리한다.
Error Event ID	에러 타입의 이벤트 아이디이다.
Ignore	해당 액티비티에서 에러 이벤트 사용 여부를 설정한다.

다음은 **[Add]** 버튼을 클릭하는 경우 나타나는 화면에 대한 설명이다.

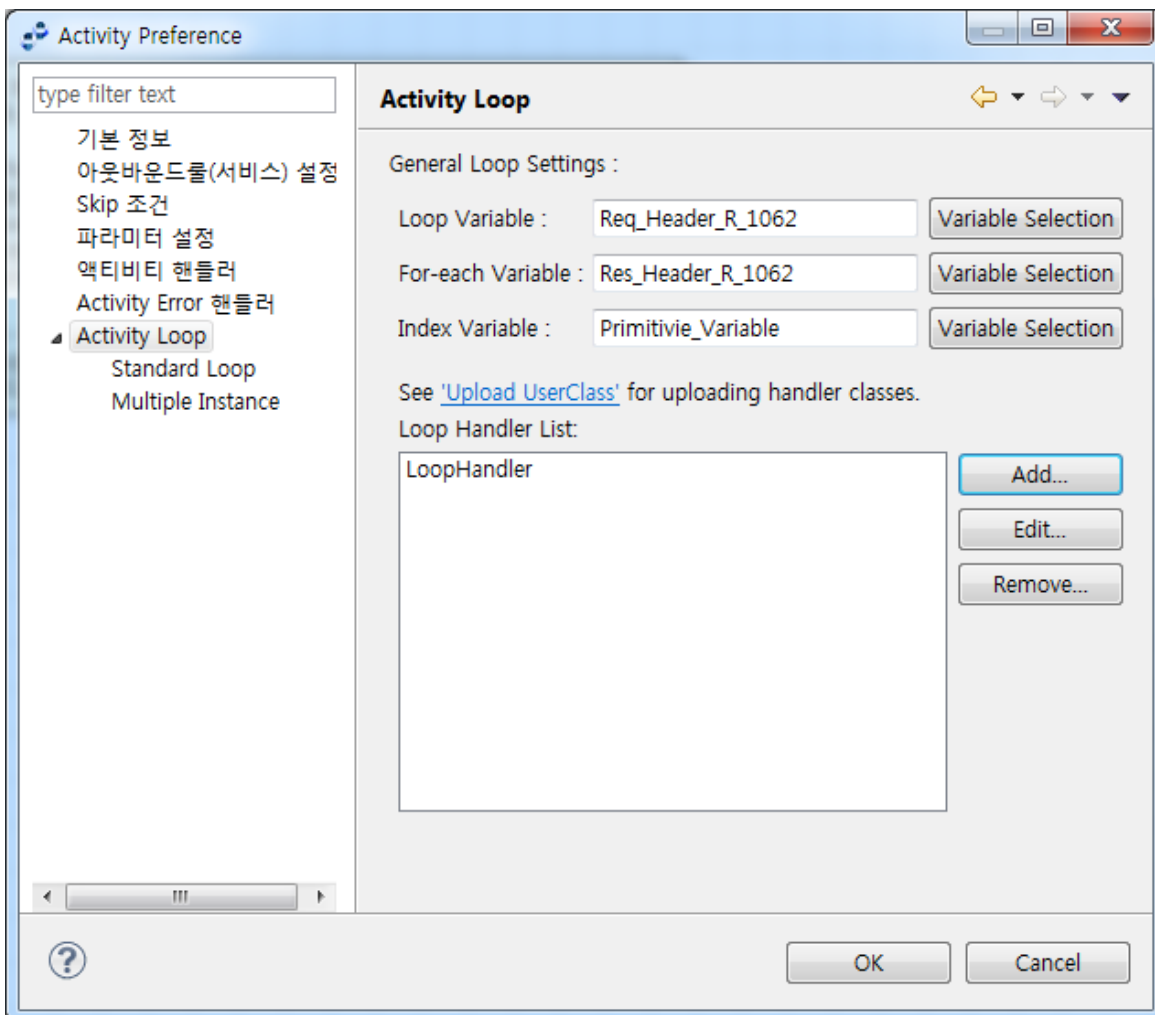


Add New Exception Handler

항목	설명
Catch All Exceptions	모든 Exception을 Catch한다.
Error Code	Catch할 에러 코드를 입력한다. Catch All Exception이 체크되어 있을 경우 자동으로 All이 된다.
Error Event	에러 이벤트를 검색하여 입력한다. [검색] 버튼을 클릭하면 Service List Dialog 에서 검색 가능하다.
Handler Class	핸들러 클래스를 검색하여 입력한다. [검색] 버튼을 클릭하면 Service List Dialog 에서 검색 가능하다.

Activity Loop

다음은 **Activity Preference 화면의 [Activity Loop]**에 대한 설명이다.



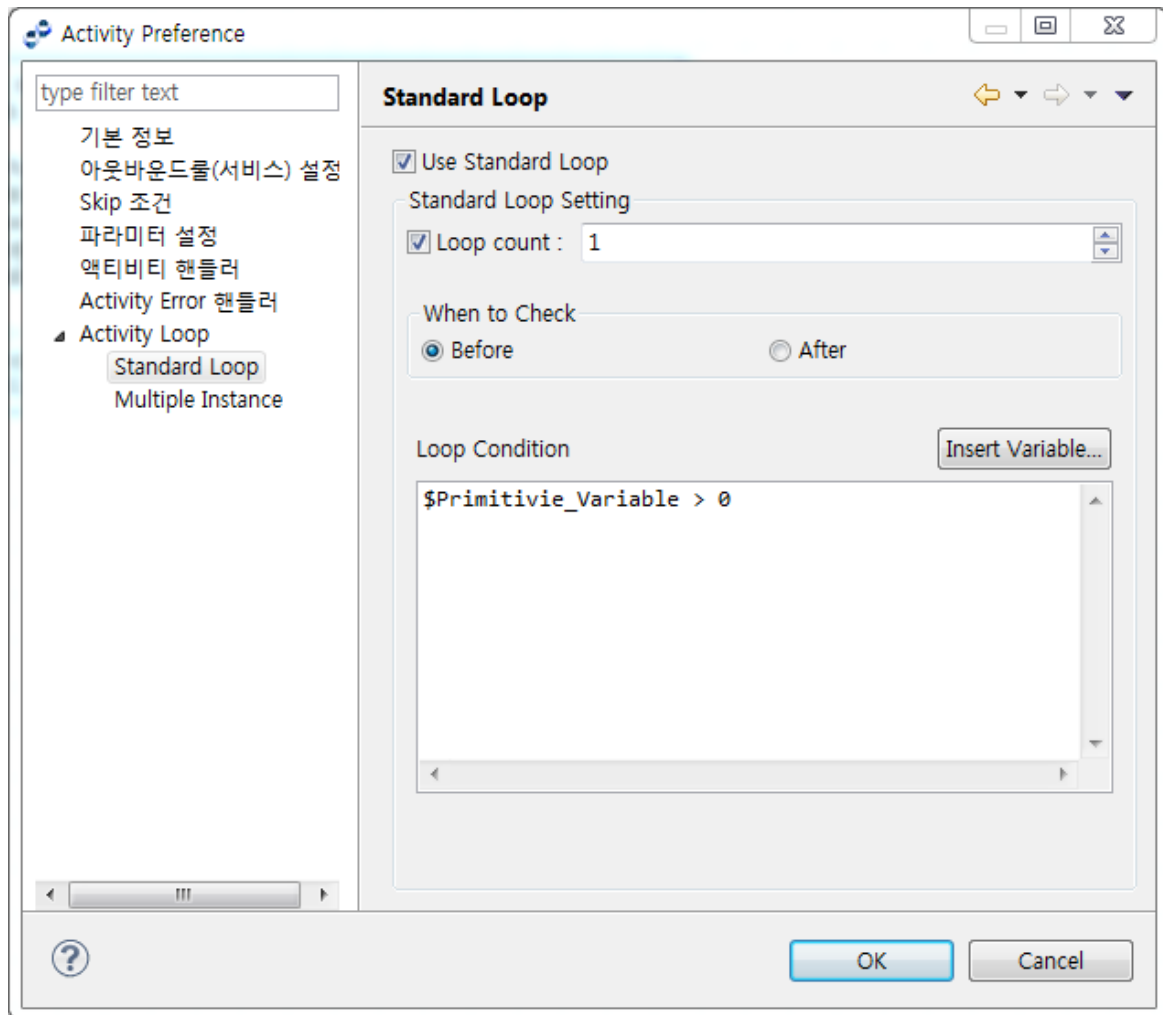
Activity Preference 화면 - Activity Loop

항목	설명
Loop Variable	Array 항목을 가진 변수이다.

항목	설명
For-each Variable	Loop 변수의 타입을 자신의 타입으로 가지는 변수이어야 한다. Loop 변수의 배열 순서대로 값이 매핑된다.
Index Variable	반복될 때 마다 하나씩 증가하는 Integer 변수이다.
Loop Handler List	Loop에서 사용할 수 있는 Handler Class를 등록할 수 있다.

• [Standard Loop]

순차적으로 일정 횟수만큼 단순 반복한다.



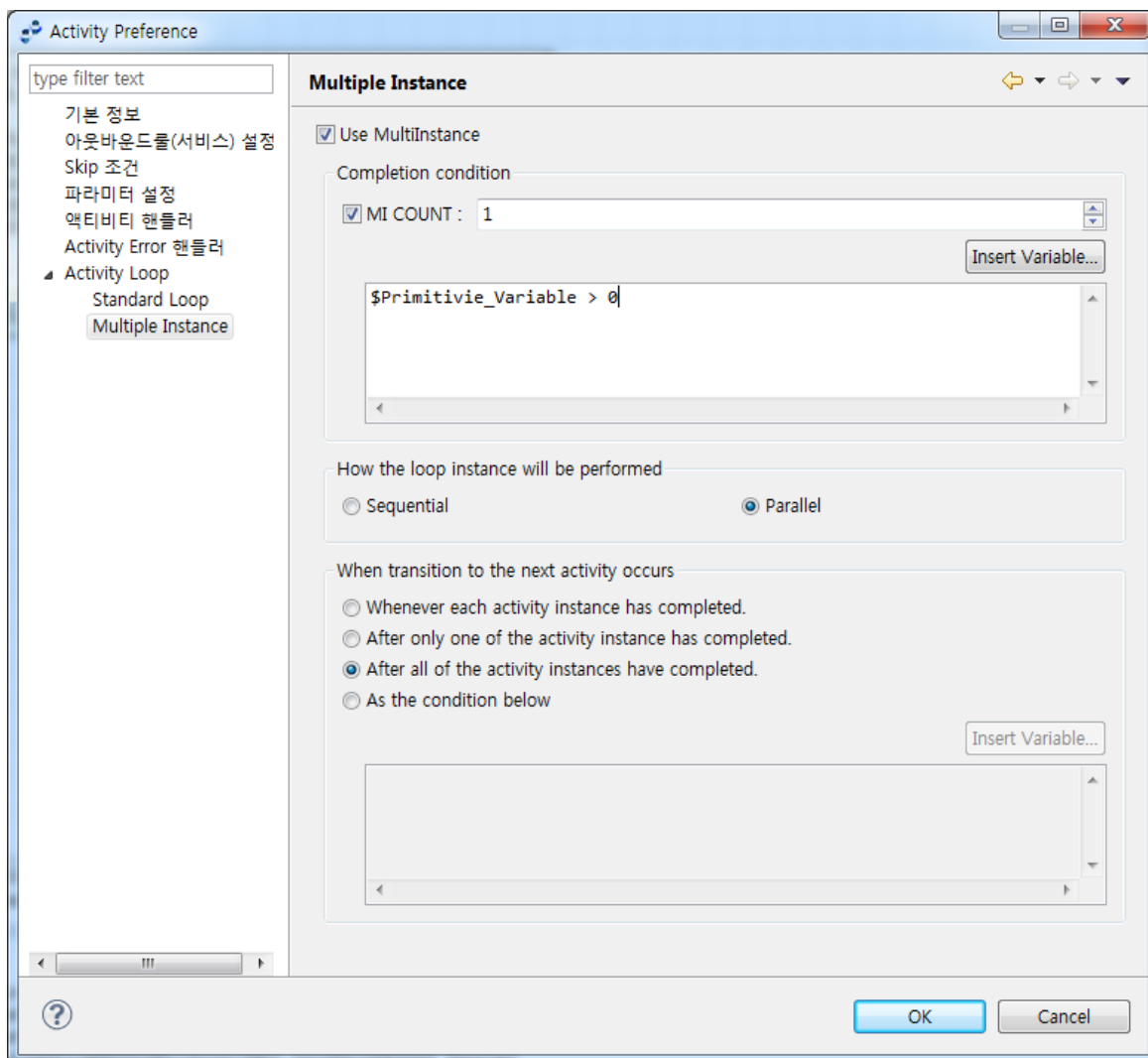
Activity Preference 화면 - Activity Loop - Standard Loop

항목	설정
Loop Count	현 액티비티의 최대 반복 횟수를 설정한다. ' Loop Condition '을 만족하지 못하는 경우 최대 반복 횟수를 모두 채우지 못하고 Loop가 종료될 수도 있다. ' Loop Count '를 설정하고 ' Loop Condition '을 설정하지 않았을 경우, 내부적으로는 ' Loop Condition '을 참으로 설정한다.

항목	설정
When to Check	<p>'Loop Condition'을 체크하는 시점을 설정한다.</p> <ul style="list-style-type: none"> • Before : 액티비티를 수행하기 전에 체크(while loop) 한다. • After : 액티비티 수행 후에 체크(until loop)한다. 'Loop Condition'을 만족하지 못할 경우라도 최소한 1회는 수행된다.
Loop Condition	<p>Loop의 수행 조건을 설정한다.</p> <p>조건이 참일 경우에만 Loop은 'Loop Count'의 수 만큼 반복 수행되며 거짓이면 Loop은 종료된다. 'Loop Count'가 설정되어 있지 않으면 'Loop Condition'을 만족하지 않을 때까지 무한 반복하게 된다.</p> <p>아무 것도 입력하지 않으면 항상 참으로 인식하지만 'Loop Count'가 설정되지 않았을 경우 'Loop Condition'은 반드시 항상 참이 아닌 값으로 설정되어 있어야 한다. 입력 창에서 <Ctrl>+<Space>로 변수를 쉽게 입력할 수도 있다.</p>

• [Multiple Instance]

순차적 또는 병렬적으로 수행되며 수행 횟수나 종료 조건도 일정하지 않은 경우가 많다.



Activity Preference 화면 - Activity Loop - Multiple Instance

- **Completion condition**

현 액티비티를 얼마나 수행할 것인가를 설정한다. '**MI Count**'로 최대 반복 횟수를 설정하고, '**Loop Condition**'으로 종료 조건을 설정한다.

- **How the loop instance will performed**

항목	설명
Sequential	인스턴스가 순차적으로 실행된다.
Parallel	모든 인스턴스가 동시에 실행된다.

- **When transition to the next activity occurs**

현 액티비티가 언제 종료 처리되어 Process Instance의 토큰이 다음 액티비티로 나갈 것인지를 설정한다.

항목	설명
Whenever each activity instance has completed	하나의 액티비티 인스턴스가 종료될 때마다 다음 액티비티로 토큰을 넘긴다. 하나 이상의 토큰이 계속 전달된다.
After only one of the activity instance has completed	하나의 액티비티 인스턴스만 종료되어도 다음 액티비티로 토큰을 넘긴다. 뒤에 수행되는 인스턴스의 토큰은 무시된다.
After all of the activity instance have completed	모든 액티비티 인스턴스가 종료되었을 때 다음 액티비티로 토큰을 넘긴다.
As the condition below	표시된 조건에 따라 그 조건이 만족되면 토큰을 넘긴다.



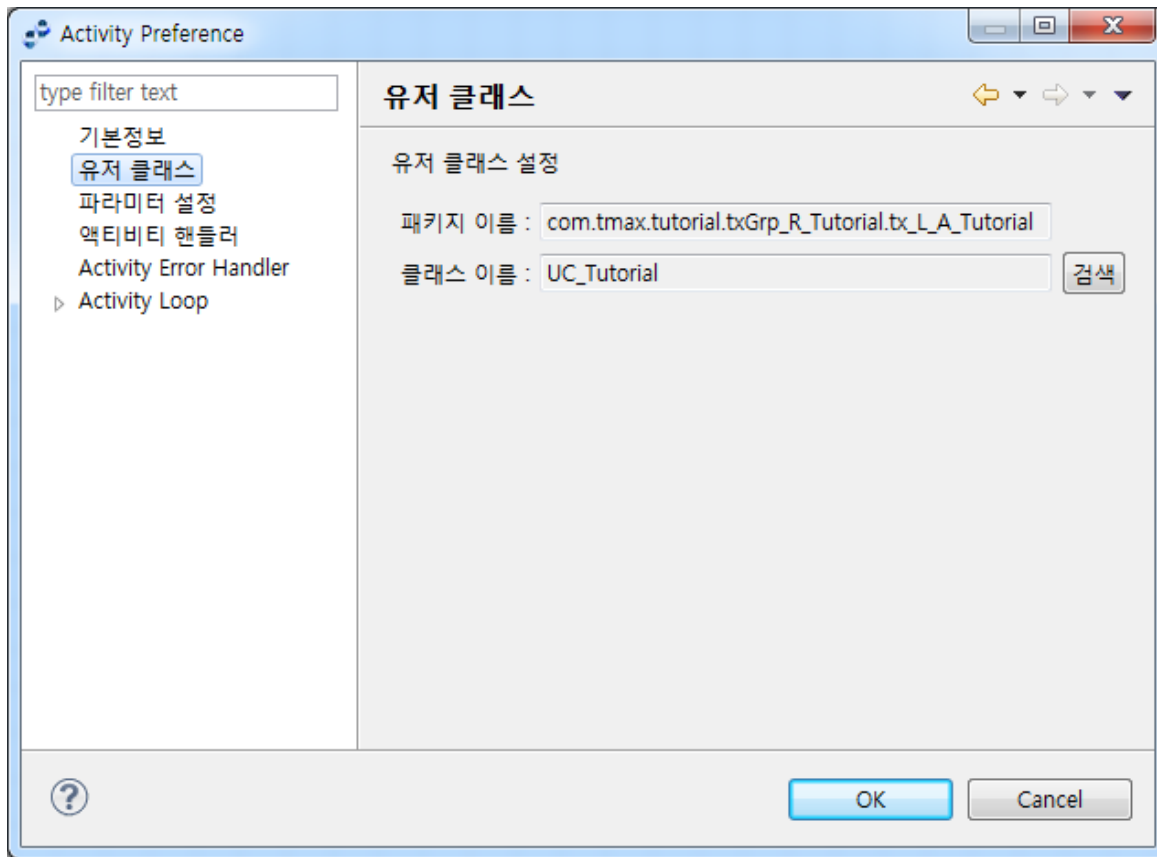
멀티인스턴스(Multiple Instance)를 사용하고 '**How the loop instance will be performed**'을 'Parallel'로 지정하는 경우 For-each 변수와 Index 변수는 동시성 이슈가 있으므로 반드시 local scope의 변수를 사용해야 한다. 하지만, '**How the loop instance will be performed**'을 'Sequential'로, '**When transition to the next activity occurs**'을 'After all of the activity instance have completed'로 지정하면 Standard Loop를 사용하는 것과 동일하게 동작한다.

6.4.2. 특정 액티비티 설정

본 절에서는 특정 액티비티를 설정하는 방법에 대한 설명한다.

유저 클래스 액티비티

유저 클래스 액티비티의 경우 **Activity Preference 화면**에서 복잡한 로직을 처리하기 위해 유저가 직접 생성한 유저 클래스 정보를 설정한다.

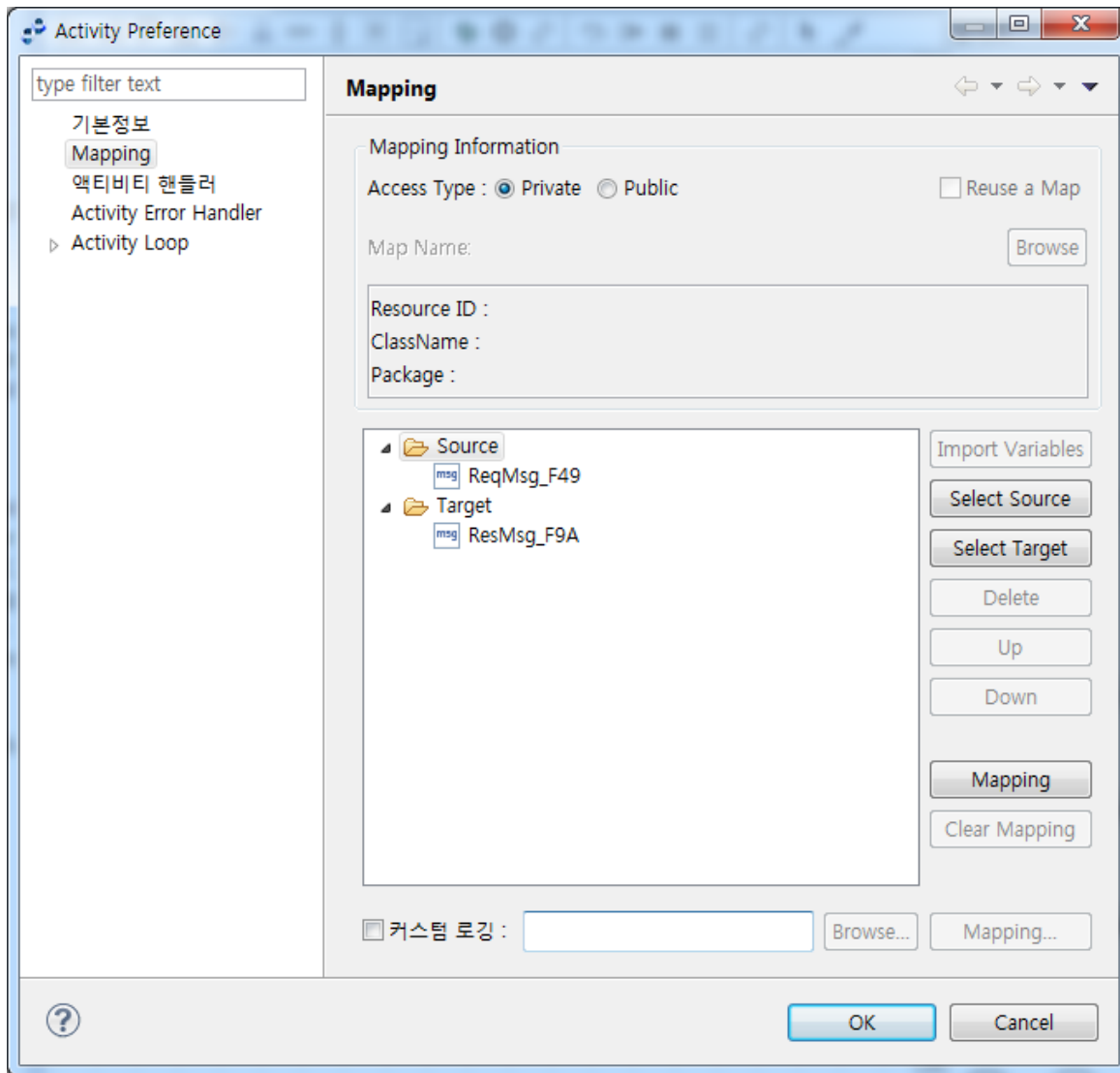


Activity Preference 화면 - 유저 클래스 액티비티

항목	설명
패키지 이름	유저 클래스의 패키지 이름이다.
클래스 이름	유저 클래스의 클래스 이름이다.

매핑 액티비티

이미 정의된 맵을 사용하여 변수들의 값을 변환한다. 매핑 액티비티의 경우 **Activity Preference** 화면에서 **[Mapping]**을 클릭해서 정보를 설정한다.



Activity Preference 화면 - Mapping

• Mapping Information

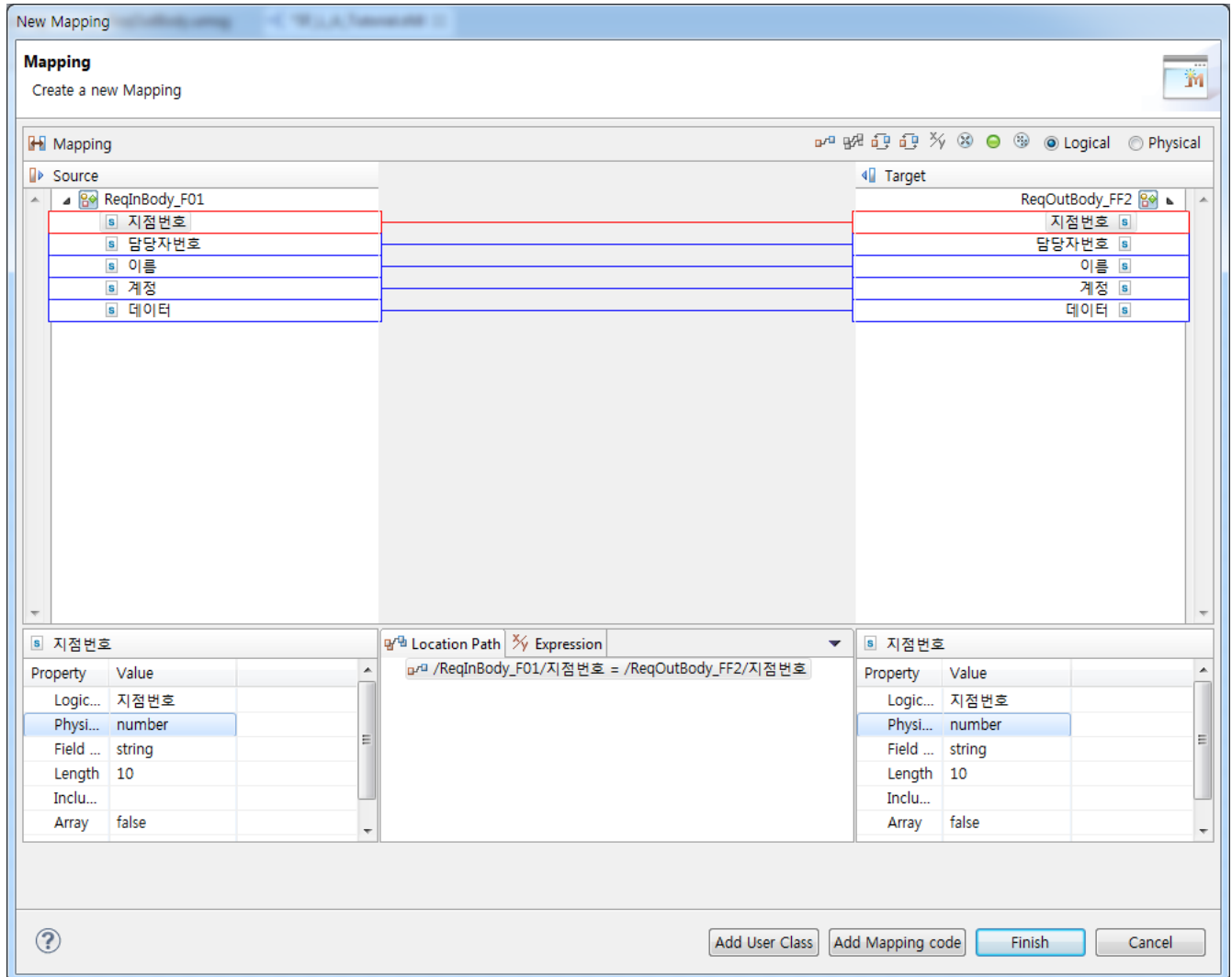
항목	설명
Access Type	Private를 선택하면 공유가 불가능하도록 액티비티 내에 정보가 저장된다.

• 버튼

버튼	설명
[Import Variables]	Public Mapping에 설정된 메시지를 import한다.
[Select Source]	Source에 해당하는 프로세스 변수를 선택할 수 있다.
[Select Target]	Target에 해당하는 프로세스 변수를 선택할 수 있다.
[Delete]	선택된 변수를 제거한다.
[Up]	변수를 한 칸 위로 이동시킨다.
[Down]	변수를 한 칸 아래로 이동시킨다.
[Mapping]	Source와 Target에 해당되는 프로세스 변수를 매핑시켜주는 Mapping Dialog 가 출력된다.

버튼	설명
[Clear Mapping]	생성된 매핑을 제거한다.

다음은 Source와 Target에 해당되는 프로세스 변수를 매핑시켜주는 **Mapping Dialog 화면**에 대한 설명이다. **[Add User Class]** 버튼을 클릭해서 유저 매핑 핸들러를 등록하거나 **[Add Mapping Code]** 버튼을 클릭해서 매핑할 때 호출될 Java Code를 입력한다.

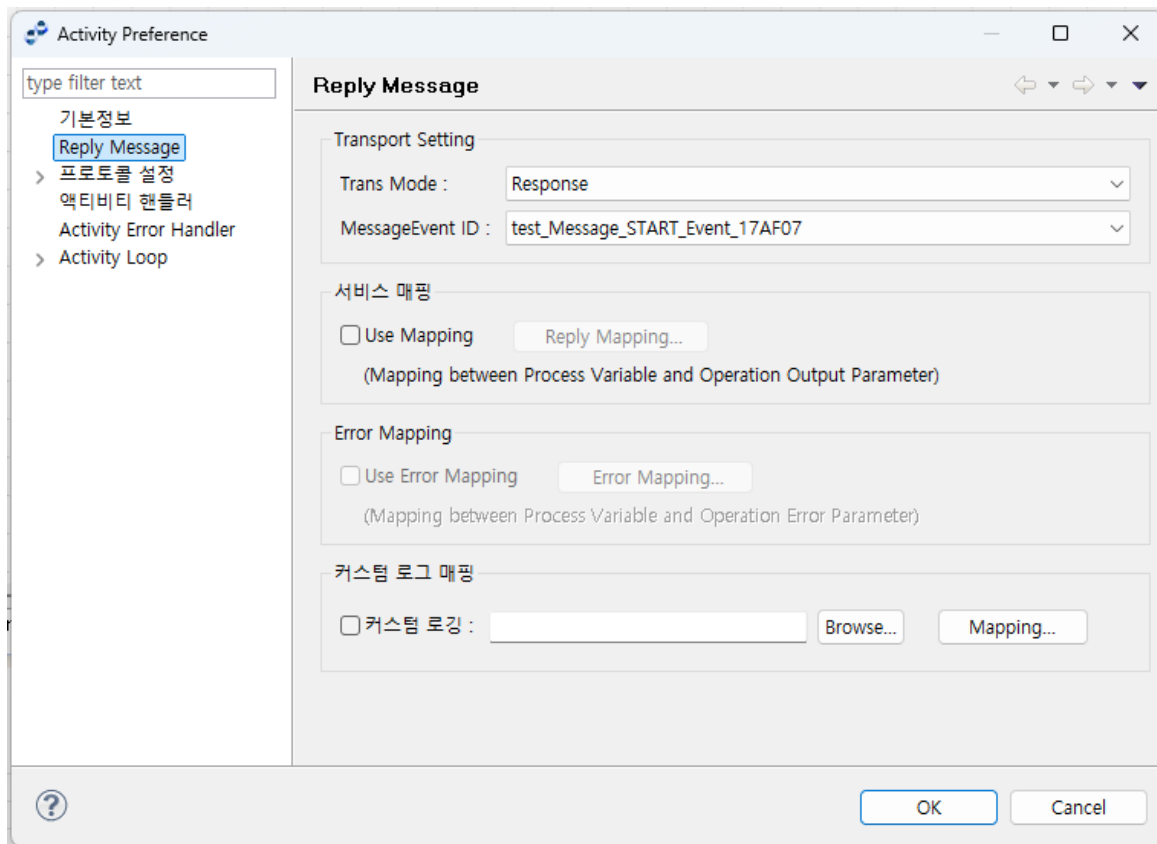


Mapping Dialog

항목	설명
Source	매핑에 사용될 변수들의 목록이다.
Target	매핑을 통해 값을 넣을 변수들의 목록이다.
Properties	Source/Target에서 선택한 변수나 필드의 정보를 조회한다. 왼쪽 Properties는 Source 영역이고, 오른쪽 Properties는 Target 영역이다.
Location Path	매핑 연결에 대한 연결 정보 값이다.
Expression	표현식을 통해 값을 매핑한다. 문자열은 큰따옴표(" ")로 감싸서 입력하고, 숫자는 그대로 입력한다.

응답 메시지 액티비티

메시지 이벤트로 들어온 인바운드 서비스에 응답을 준다. 응답 메시지 액티비티의 경우 **Activity Preference** 화면에서 **[Reply Message]**를 선택해서 정보를 설정한다.



Activity Preference 화면 - Reply Message

항목	설명
Trans Mode	<ul style="list-style-type: none">◦ Response : 응답 메시지 액티비티를 응답(Response) 용도로 사용할 수 있다.◦ Error : 응답 메시지 액티비티를 에러 응답 메시지 용도로 사용할 수 있다.◦ Response None : 응답 메시지를 보내지 않을 수 있다.
MessageEvent ID	서비스 플로우에 있는 메시지 이벤트를 설정할 수 있다.
Use Mapping	<p>'Trans Mode'가 Response일 때 사용 가능하며 Reply Mapping 화면에서 프로세스 변수와 Output 파라미터 간에 매핑을 설정한다.</p> <p>[Reply Mapping] 버튼을 클릭하면 매핑 다이얼로그가 나타난다.</p>
Use Error Mapping	<p>'Trans Mode'가 에러일 때 사용 가능하며 Error Mapping 화면에서 프로세스 변수와 에러 파라미터 간에 매핑을 설정한다.</p> <p>[Error Mapping] 버튼을 클릭하면 매핑 다이얼로그가 나타난다.</p>

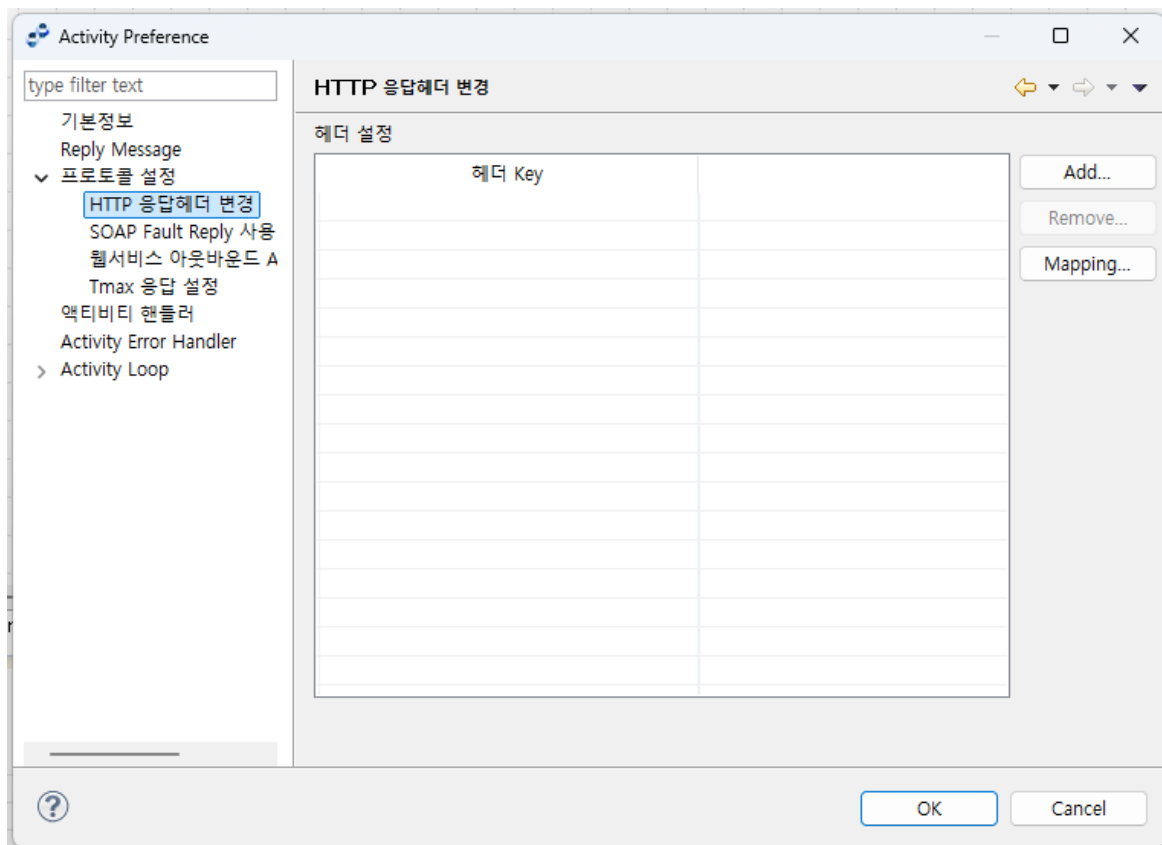
항목	설명
커스텀 로그 매핑	<p>해당 액티비티에 설정된 커스텀 로그와 액티비티 변수를 매핑한다.</p> <ul style="list-style-type: none"> ◦ [Browse...] 버튼 : 커스텀 로그 룰을 검색한다. 해당 버튼을 누르면 Service List Dialog에서 검색 가능하다. ◦ [Mapping...] 버튼 : 커스텀 로그 룰의 요청메시지를 매핑한다.

프로토콜 설정

응답 메시지 설정 시 특정 프로토콜에서 사용되는 설정을 적용할 수 있다.

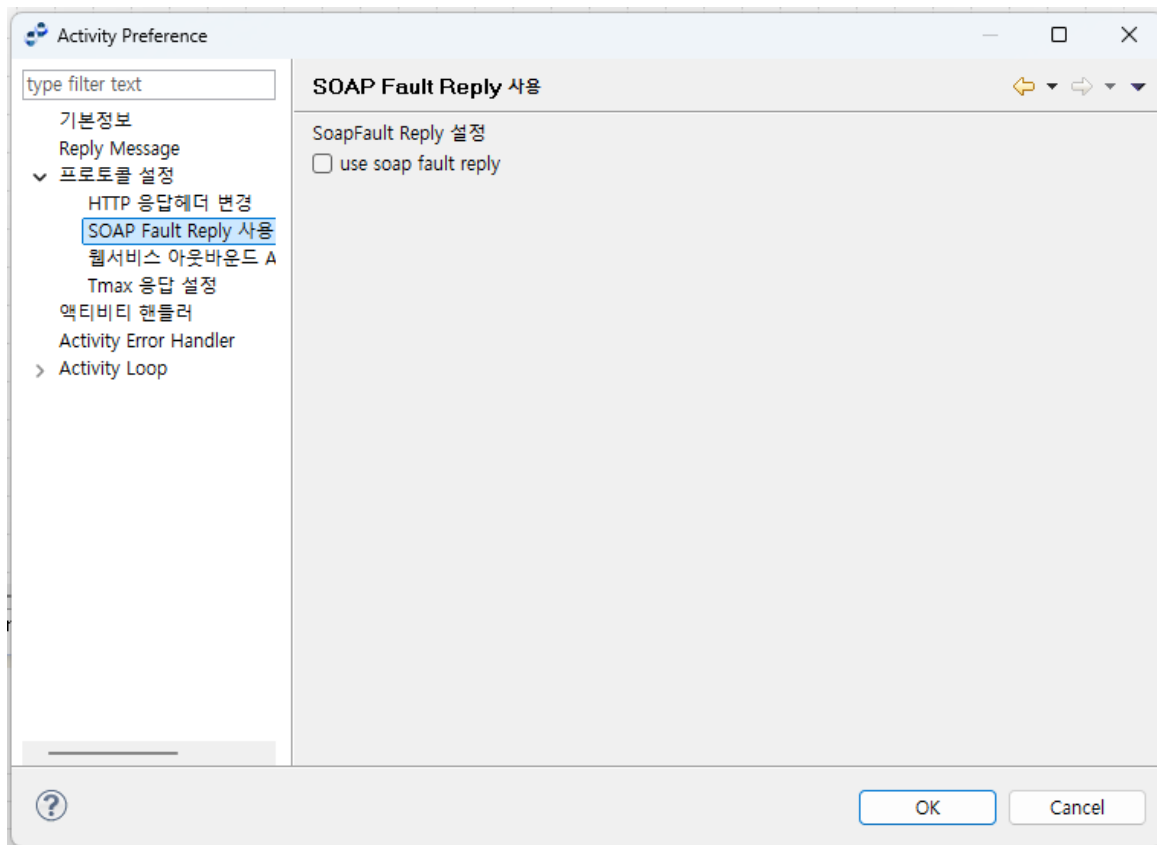
• HTTP 응답 헤더 변경

응답 메시지의 HTTP 헤더를 동적으로 변경한다.



프로토콜 설정 화면 - HTTP 응답 헤더 변경

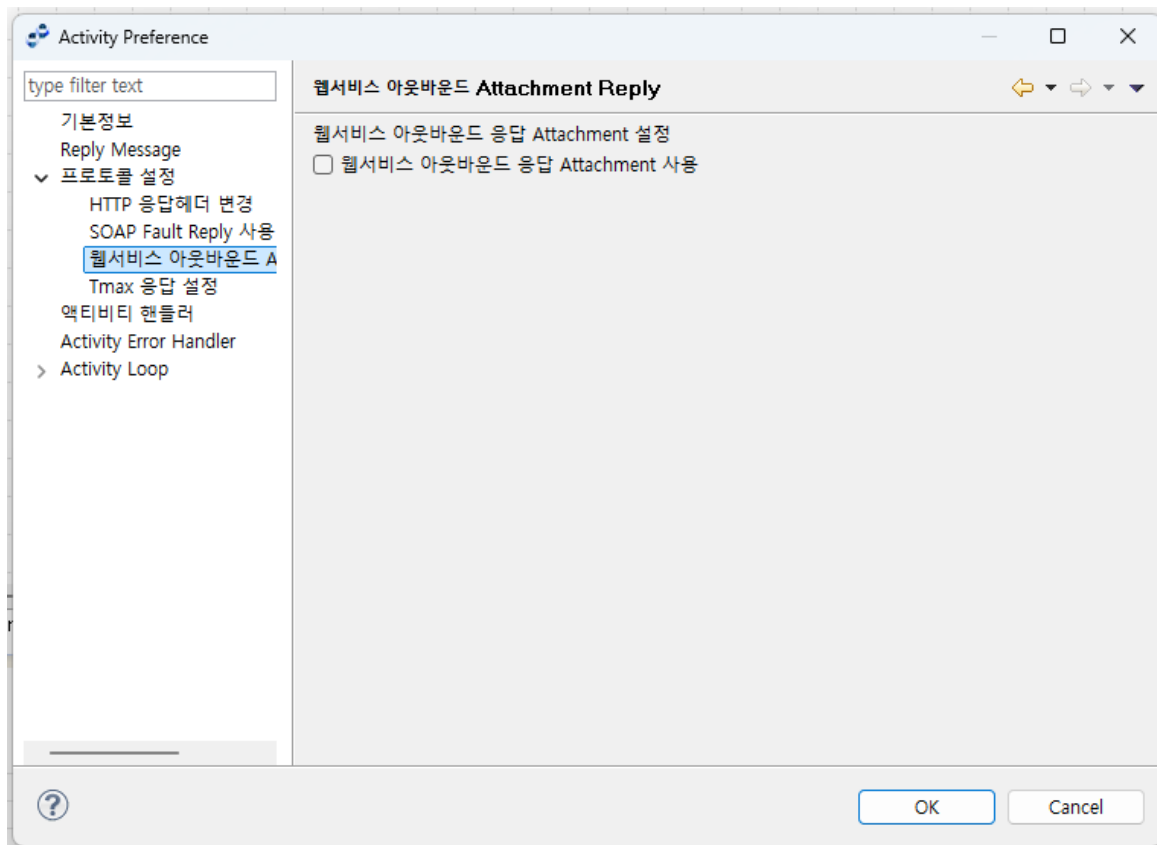
• SOAP Fault Reply 사용



프로토콜 설정 화면 - SOAP Fault Reply 사용

항목	설명
use soap fault reply	[거래] > [업무오류 응답 메시지]에 지정한 메시지를 soap fault 스펙에 맞는 fault 메시지로 변환한다.

• 웹서비스 아웃바운드 Attachment Reply



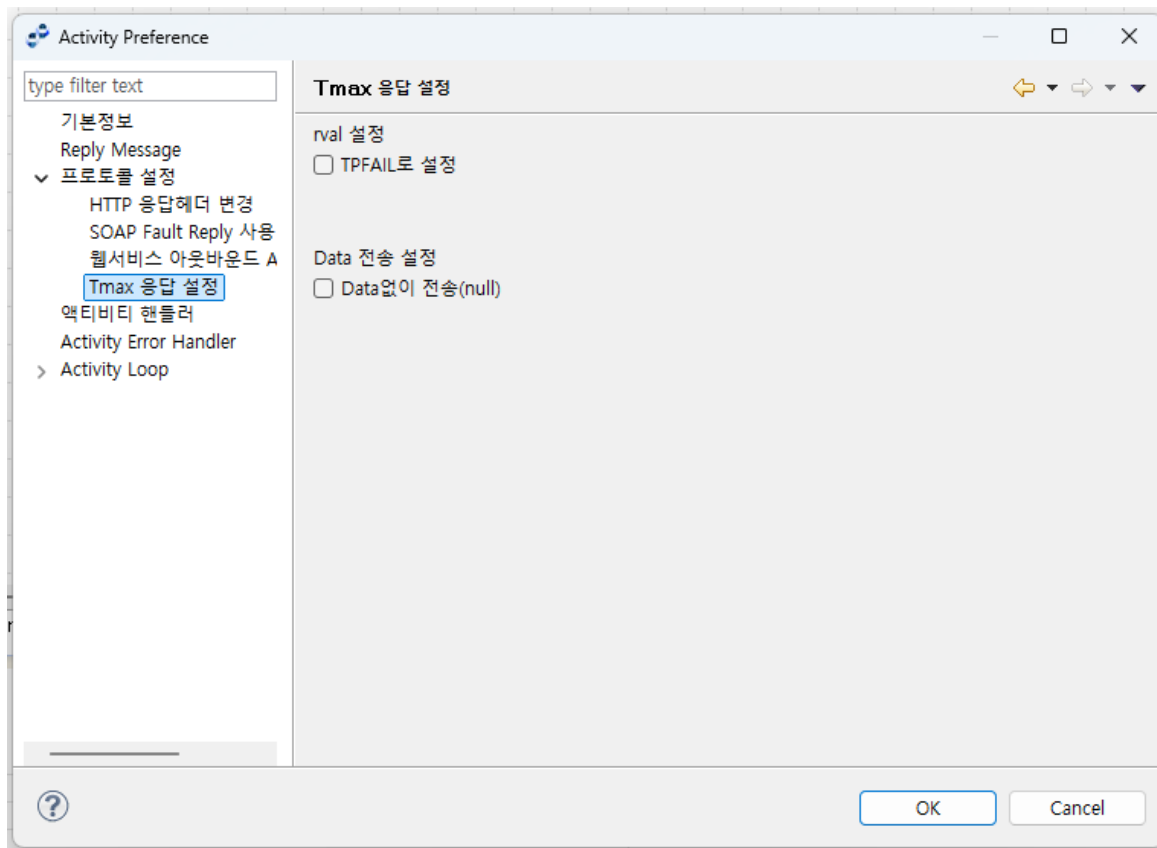
프로토콜 설정 화면 - 웹서비스 아웃바운드 Attachment Reply

항목	설명
웹서비스 아웃바운드 응답 Attachment 사용	웹서비스 응답 메시지에 첨부 파일을 설정하여 전송한다.

• Tmax 응답 설정

응답 메시지의 반환 값과 전송 데이터의 처리 방식을 설정한다.

설정이 적용된 응답은 [모니터링] > [트랜잭션 트레이스]의 응답 메시지에서 확인할 수 있고, 설정이 적용되지 않은 응답은 기존 방식으로 로깅된다. 설정에 의해 메시지가 변경된 경우 기존 메시지는 트레이스 로그의 ORIGINALMESSAGE 항목에 기록된다. 단, 설정 사용 시 메시지 로깅 방식이 변경되므로 [FEP] > [응답 송신] 기능은 사용할 수 없다.



프로토콜 설정 화면 - Tmax 응답 설정

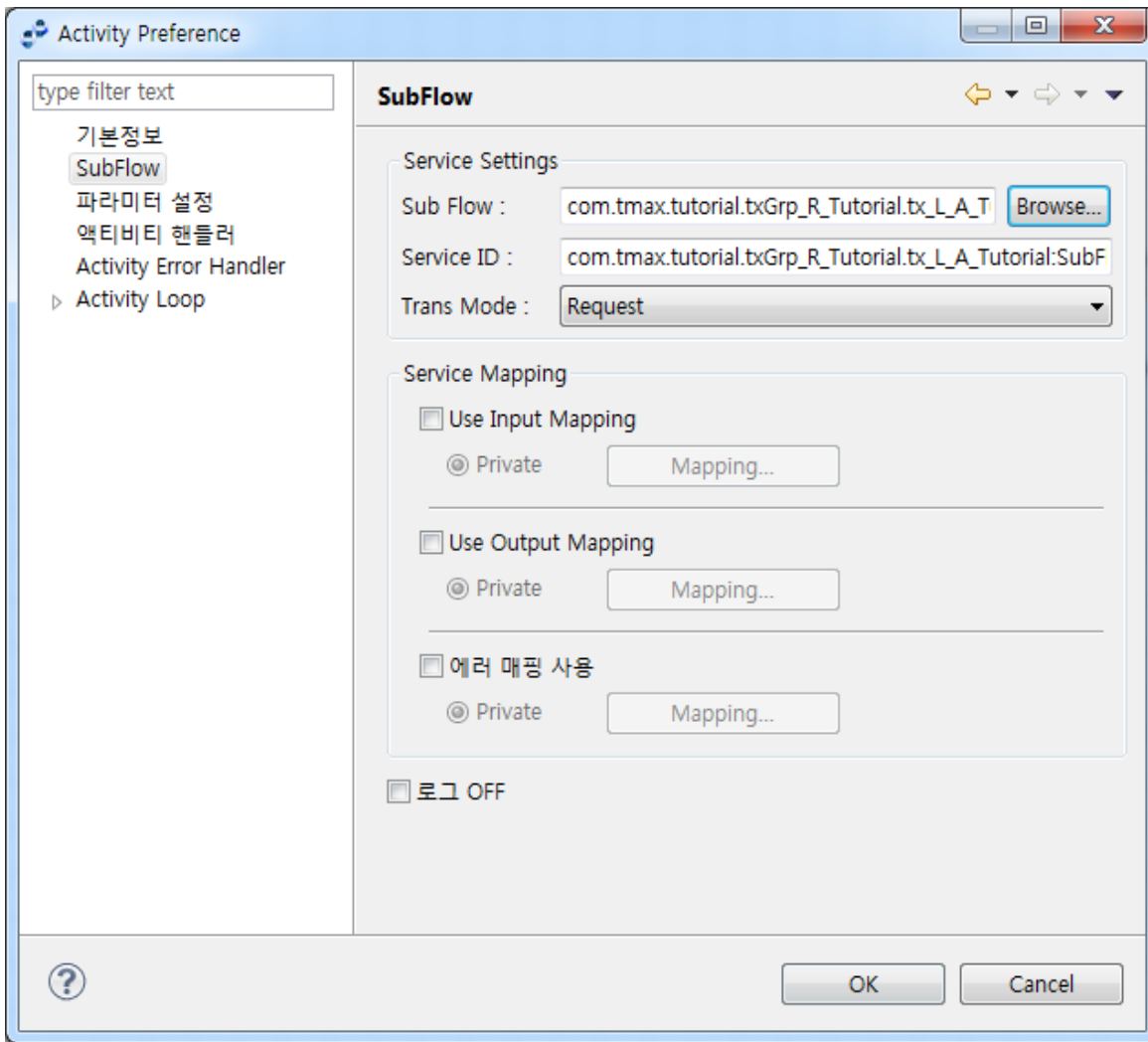
항목	설명
TPFAIL로 설정	응답 메시지의 rval이 TPFAIL로 설정된다.
Data없이 전송(null)	응답 메시지의 데이터가 null로 설정된다.

다음은 설정에 따른 로그 기록 방식이다.

설정 적용	응답 메시지 로그
TPFAIL, 데이터 없음(null)	Reply[TPFAIL] Data[null]
TPFAIL, 데이터 있음	Reply[TPFAIL] Data[기존 메시지]
TPSUCCESS, 데이터 없음(null)	Reply[TPSUCCESS] Data[null]
TPSUCCESS, 데이터 있음	기존 메시지

SubFlow

응답 메시지 액티비티의 경우 **Activity Preference 화면**에서 다른 플로우를 호출하기 위해 **[SubFlow]**를 선택해서 정보를 설정한다.



Activity Preference 화면 - SubFlow







항목	설명
Sub Flow	사전에 생성한 서브 서비스 플로우(또는 서비스 플로우)를 설정할 수 있다.
Service id	설정한 서브 플로우의 Service ID이다.
Trans Mode	<ul style="list-style-type: none"> Request : 가장 일반적인 메시징 형태로 요청에 대한 응답을 기대하는 전송모드이다. One Way : 응답을 기다리지 않고 요청 메시지만 전달하는 전송모드이다. One Way with ack : 요청 메시지가 처리된 후 ACK 메시지를 기대하는 형태이다. 보통 딜리버리 채널에서 전달 보장을 처리한 후 ACK 메시지 또는 NAK 메시지를 답변으로 보내는 전송모드이다.
Service Mapping	아웃바운드 서비스의 변수와 Activity Parameter 사이에 in/out 매핑이 가능하다. 매핑을 만들 때 실제 맵 파일이 생성되도록 하는 'public' 설정과 SFDL 내 서비스 매핑에 정보를 숨겨두는 'private' 설정이 있다. 즉, 'private' 매핑의 경우는 다른 플로우나 액티비티에서 재사용할 수 없다.

멀티바인딩(MultiBinding) 액티비티

여러 서비스를 하나의 액티비티에서 처리하기 위해 생성하는 액티비티로 멀티바인딩에 설정된 서비스를 호출한다.

6.5. 이벤트

다음은 이벤트의 종류에 대한 설명이다.

항목	설명
 (메시지 이벤트)	메시지 이벤트로 메시지를 수신하는 이벤트이다. Start와 Intermediate 타입을 지원한다.
 (타이머 이벤트)	시작, 종료시간, 기간을 설정하여 특정 주기로 프로세스를 시작할 수 있다. Intermediate 타입을 지원한다.
 (에러 이벤트)	에러가 발생하는 경우 수행되는 이벤트이다. Start, Intermediate, End 세 가지 타입이 모두 지원된다.
 (링크 이벤트)	동기화나 프로세스 흐름 제어를 위해 사용한다. Start, Intermediate, End 세 가지 타입이 모두 지원된다.
 (정상 종료 이벤트)	정상 종료를 나타내는 이벤트이다. End 타입만을 지원한다.
 (비정상 종료 이벤트)	강제 종료 이벤트로서 프로세스 전체를 종료시킨다. 이 이벤트를 통해 종료될 때 WebAdmin에서 비정상 종료로 인식한다. End 타입만을 지원한다.

6.5.1. 이벤트 설정

이벤트를 에디터 화면으로 갖고 온 뒤 더블클릭 또는 객체 위로 마우스를 올리면 나타나는 **Event Preference 화면**에서 이벤트를 설정한다.



이벤트 설정

Event Preference 화면에서 이벤트의 공통적인 기본 정보 설정을 한다. Event Class를 제외한 나머지 항목은 **Activity Preference 화면**과 같으므로 생략한다.

Event Preference 화면 - 기본 정보 화면

항목	설명
Event Class	이벤트 객체의 역할과 서비스 플로우에서의 위치에 따라 Start, Intermediate, End 타입을 지정할 수 있다. 몇몇 이벤트의 경우 제한이 있기도 하다.
커스텀 로깅	커스텀 로깅을 사용한다. <ul style="list-style-type: none"> ◦ [Browse...] 버튼 : 커스텀 로그 아웃바운드 룰을 설정한다. ◦ [Mapping...] 버튼 : 매핑을 설정한다.
세션 ID 사용	플로우 내부에서 세션을 관리하기 위해 아이디를 설정하려면 [New] 버튼을 클릭해서 세션 아이디를 직접 입력이 가능하고, [Search] 버튼을 선택할 때 텍스트박스 옆의 [Search] 버튼을 클릭하면 세션 선택 다이얼로그에서 세션을 아이디를 설정할 수 있다.

6.5.2. 타입별 이벤트 설정

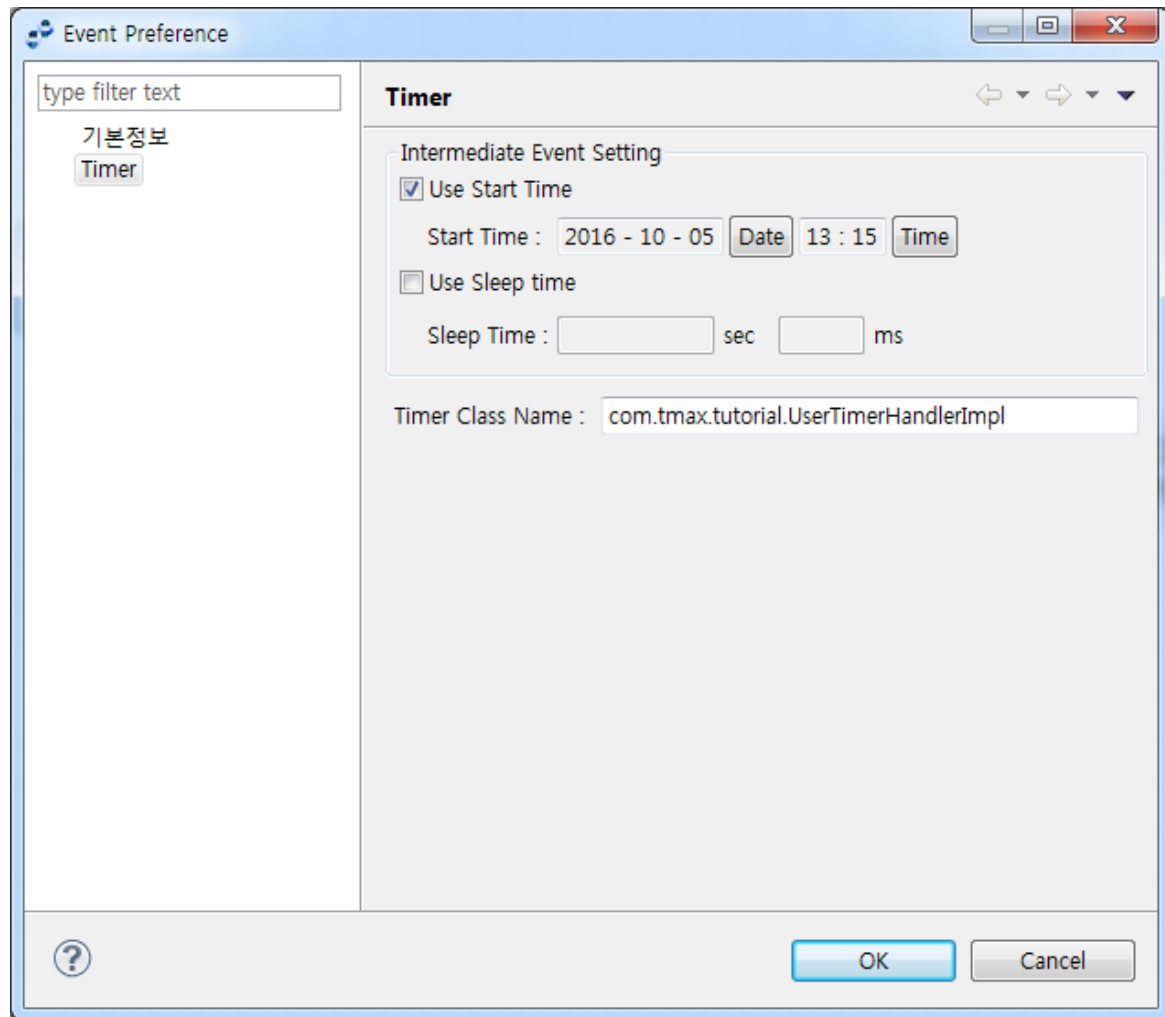
• 메시지 이벤트

메시지 이벤트의 경우 **파라미터 설정** 항목이 있으며, 항목에 대한 설명은 [Activity Preference 화면](#)의 파라미터 설정과 같으므로 생략한다.

• 타이머 이벤트

타이머 이벤트의 경우 타이머 정보를 설정한다. **Timer** 항목에서 시작,종료시간, 기간을 설정하여 특정 주기로

프로세스를 시작할 수 있다.



Event Preference 화면 - Timer

항목	설명
Intermediate Event Setting	<ul style="list-style-type: none">• 'Start time'만 설정한 경우 : 설정된 시작 시간 이전까지 sleep하고, 만약 이후라면 영원히 sleep한다.• 'Sleep time'만 설정한 경우 : 설정한 기간 만큼만 sleep한다.• 'Start time'과 'Sleep time'을 모두 설정한 경우 : 시작 시간 이전까지 sleep하고, 만약 이후라면 start time+n*sleep time까지 sleep한다.
Timer Class Name	유저 클래스를 직접 등록하여 사용이 가능하다.

• 에러 이벤트

에러 이벤트의 경우 에러 코드 정보를 설정한다. 에러 코드를 입력해야 한다. AnyLink에서 정의한 특수한 에러코드들은 Error Code 설정 옆의 ? 버튼을 통해 확인할 수 있다.

The 'Event Preference' dialog box is shown with the '기본정보' (Basic Information) tab selected. The 'Event Type' is set to 'Error'. The 'ID' field contains 'SF_L_A_Tutorial_Error_INTER_Event_0FAA1B'. The 'Name' and 'Description' fields both contain 'Error'. The 'Error Code' field has a small icon and a help button. The 'Event Class' section has three radio buttons: 'Start', 'Intermediate' (which is selected), and 'End'. At the bottom, there are 'OK' and 'Cancel' buttons and a help icon.

Error Preference 화면 - 에러 이벤트

• 링크 이벤트









링크 이벤트의 경우 Link ID 정보를 설정한다. 연결할 상대의 아이디를 입력해야 한다.

The 'Event Preference' dialog box is shown with the '기본 정보' (Basic Information) tab selected. The 'Event Type' is set to 'Link'. The 'ID' field contains 'FlowEvent_Link_END_Event_02B11D'. The 'Name' and 'Description' fields both contain 'Link'. The 'Link ID' field contains 'LinkId'. The 'Event Class' section has three radio buttons: 'Start', 'Intermediate', and 'End' (which is selected). At the bottom, there are 'OK' and 'Cancel' buttons and a help icon.

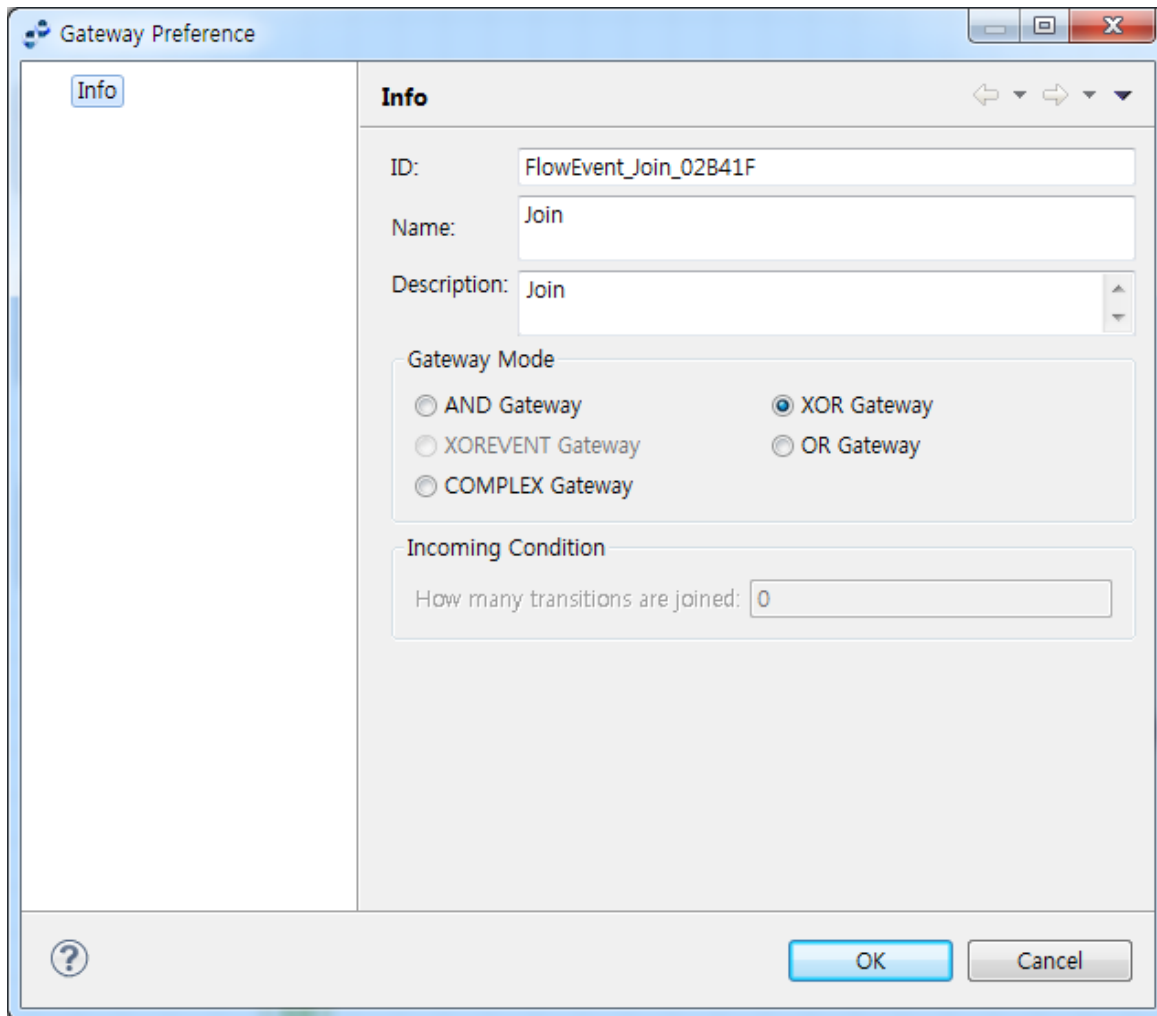
Event Preference 화면 - 링크 이벤트

6.6. 게이트웨이

다음은 게이트웨이의 종류에 대한 설명이다.

항목	설명
 (XOR Split)	XOR Split는 여러 개의 경로 중 하나의 경로로 프로세스 인스턴스가 라우팅되는 게이트웨이이다.
 (XOR Join)	XOR Split로 분기된 프로세스는 XOR Join으로 합쳐질 수 있다. XOR Split의 각 트랜지션(Transition)에 조건을 설정해야 하며, 이 조건들은 우선순위에 따라 조건을 검사하여 참(True)이면 진행한다. 만약에 만족하는 조건이 존재하지 않는 경우 런타임 예외가 발생하므로 'otherwise' 타입을 하나 지정하는 것이 좋다.
 (XOR Event Split)	<p>XOR Split와 유사하지만 지연 선택의 경우처럼 다음 수행될 액티비티 트랜지션(Activity Transition)에 의해 결정하지 않고 다음 액티비티들의 자체 발생 여부를 가지고 있는 라우팅되는 게이트웨이이다.</p> <p>XOREvent Split 다음에는 반드시 타이머나 메시지 이벤트 또는 이벤트로 시작하는 블록만 올 수 있다.</p>
 (Or Split)	Or Split는 부팅할 때의 조건에 맞는 모든 경로로 프로세스 인스턴스가 라우팅되는 게이트웨이이다.
 (Or Join)	Or Split 또한 Or Join으로 다시 합쳐져야 하며 트랜지션의 조건(Condition)을 설정하는 부분은 XOR Split와 동일하다. 다만 여러 개의 트랜지션이 겹쳐지는 조건을 가지도 있더라도 조건에 맞는 트랜지션이면 모두 수행된다는 점만 다르다.
 (And Split)	And Split는 조건 없이 모든 경로로 프로세스 인스턴스가 라우팅되는 게이트웨이이다.
 (And Join)	And Split는 모든 트랜지션으로 분기되기 때문에 각 부팅할 때의 조건(Condition)은 설정하지 않아야 한다. 비슷한 개념으로 And Join은 모든 경로에서 실행이 완료된 결과가 도착해야 다음 액티비티를 수행한다.
 (Complex Join)	Complex Join은 Join할 개수를 지정하여 지정된 개수만큼 도착하면 다음으로 진행하는 형태이다.

다음은 게이트웨이를 설정하는 **Gateway Preference 화면**에 대한 설명이다.






Gateway Preference 화면

항목	설명
Gateway Mode	선택한 게이트웨이의 종류를 표시하고 변경이 가능하다.
Incoming Condition	Complex Gateway인 경우 설정할 수 있다. 몇 개의 트랜지션으로 합칠 것인지 설정한다.

6.7. 스왑라인 / 블록 / ANNOTATIONS

다음은 스왑라인 / 블록 / ANNOTATIONS 객체에 대한 설명이다.

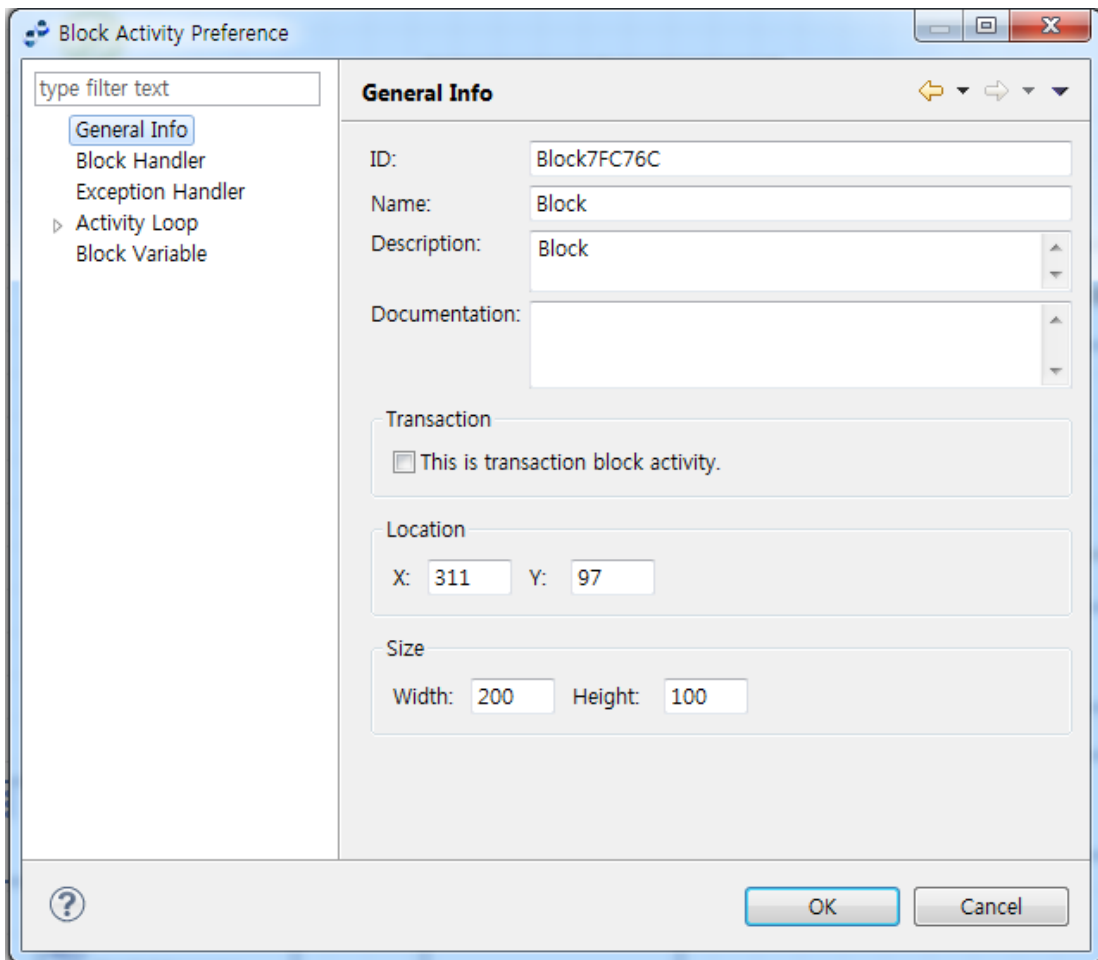
항목	설명
 (스왑라인)	스왑라인은 서비스 플로우상에 Task를 구분해주는 일종의 선이다. 스왑라인을 사용하여 Task 간의 병렬적으로 이루어지는 우선순위나 행동들을 쉽게 구분할 수 있다. 스왑라인 상단을 더블클릭하여 발생하는 다이얼로그에서 아이디, 이름, 크기 색상을 변경할 수 있다.
 (블록 액티비티)	블록 내부에서 사용할 수 있는 변수를 설정할 수 있으며, 블록 단위로 Message, Timer, Error 등의 이벤트를 설정할 수 있다.

항목	설명
 (ANNOTATIONS)	사용자의 편의를 위한 객체로 설명을 기술할 수 있다. 플로우의 메모 기능을 제공한다.

6.7.1. 블록 액티비티

특정 구역을 진행하는 도중에 발생하는 사건들이나 예외에 대한 처리를 위해서 특정 구역을 묶어서 표현하는 것을 블록(Block)이라고 한다. Java의 함수와 비슷한 개념으로 서비스 플로우 액티비티의 묶음이다.

블록에서도 액티비티와 마찬가지로 **Block Handler, Exception Handler, Activity Loop, Block Variable** 설정이 가능하다. **[Block Variable]** 메뉴에서 블록 내에서만 사용 가능한 변수를 정의할 수 있다.



Block Activity Preference 화면

항목	설명
This is transaction block activity	블록 액티비티는 트랜잭션이 지원되는 트랙잭션 블록 기능을 제공한다. 트랜잭션 블록을 사용할 경우 블록에 진입할 때 Tx_Begin, 블록을 None 이벤트로 종료할 때 Tx_Commit, 블록을 비정상 종료하거나 Terminate 이벤트로 종료할 경우 Tx_Rollback이 자동으로 호출된다.

6.8. Utility

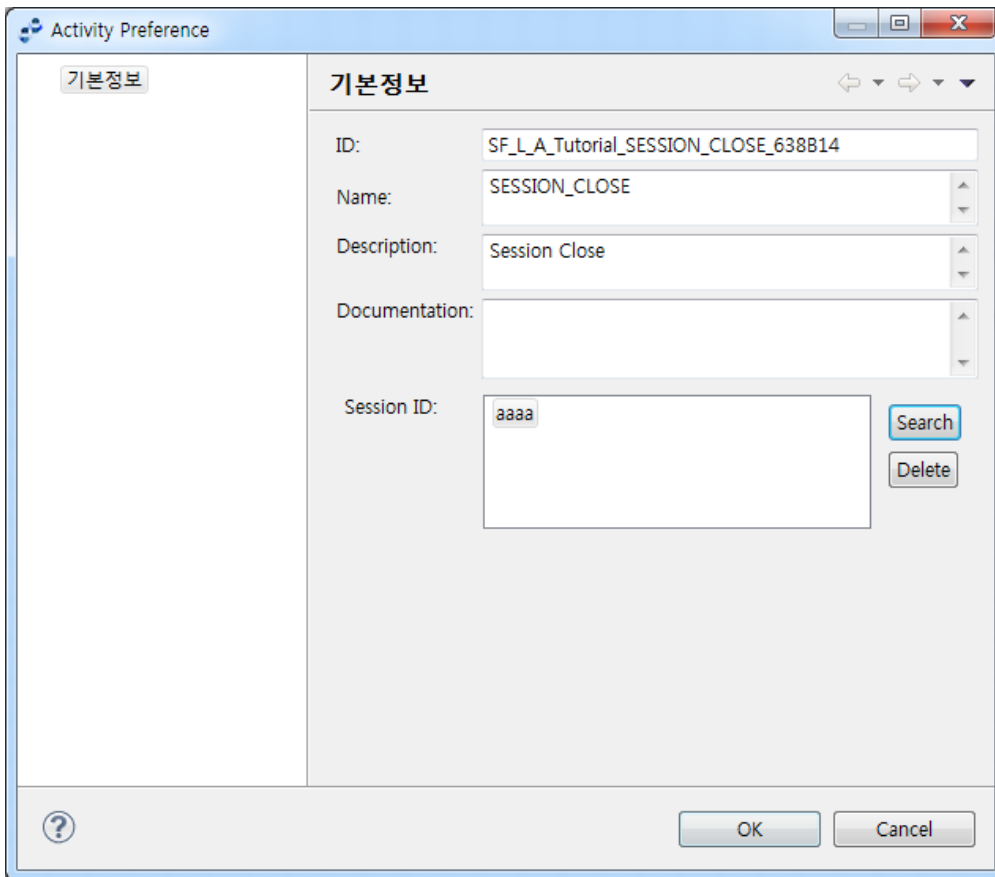
Utility는 서비스 플로우에서 사용되는 유틸리티 모음이다.

다음은 Utility 객체에 대한 설명이다.

항목	설명
 (SESSION CLOSE ACTIVITY)	Session ID를 입력하여 해당하는 세션을 닫는다.
 (SESSION ABORT ACTIVITY)	Session ID를 입력하여 해당하는 세션을 강제 종료시킨다.
 (NETWORK MANAGEMENT ACTIVITY)	망의 상태를 변경하거나 망의 상태를 조회한다.
 (MANAGER APPROVAL ACTIVITY)	단말 로그인/로그아웃이나 단말 사용자나 책임자 정보를 조회한다.
 (HTTP MULTIPART ACTIVITY)	HTTP 멀티파트로 수신된 파일을 임시경로에 저장한다. 해당 파일들은 플로우가 끝나면 자동 삭제된다.
 (BATCH PROGRESS ACTIVITY)	배치 진행률을 변경하거나 조회한다.
 (TCP MESSAGE SPLIT ACTIVITY)	다음 Intermediate Event에서 TCP 메시지를 분할해서 수신받도록 한다.
 (CONTROL SESSION ACTIVITY)	엔드포인트 ID와 세션키를 입력하여 해당하는 세션을 강제 종료시킨다.

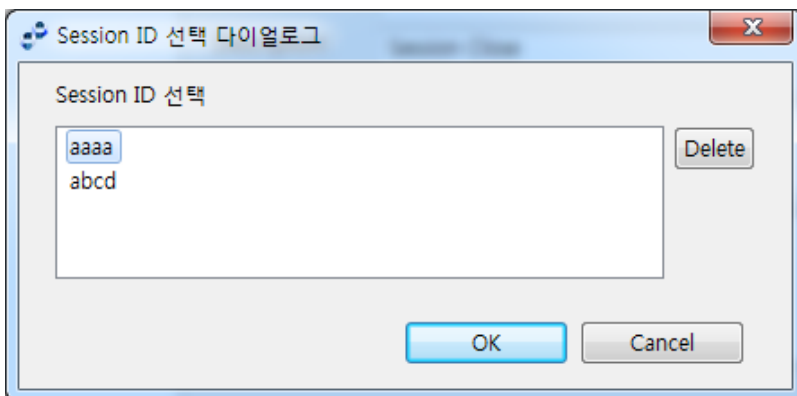
6.8.1. SESSION CLOSE ACTIVITY

다음은 SESSION CLOSE ACTIVITY의 **Session Close Preference 화면**에 대한 설명이다.



Session Close - Session Abort Preference 화면

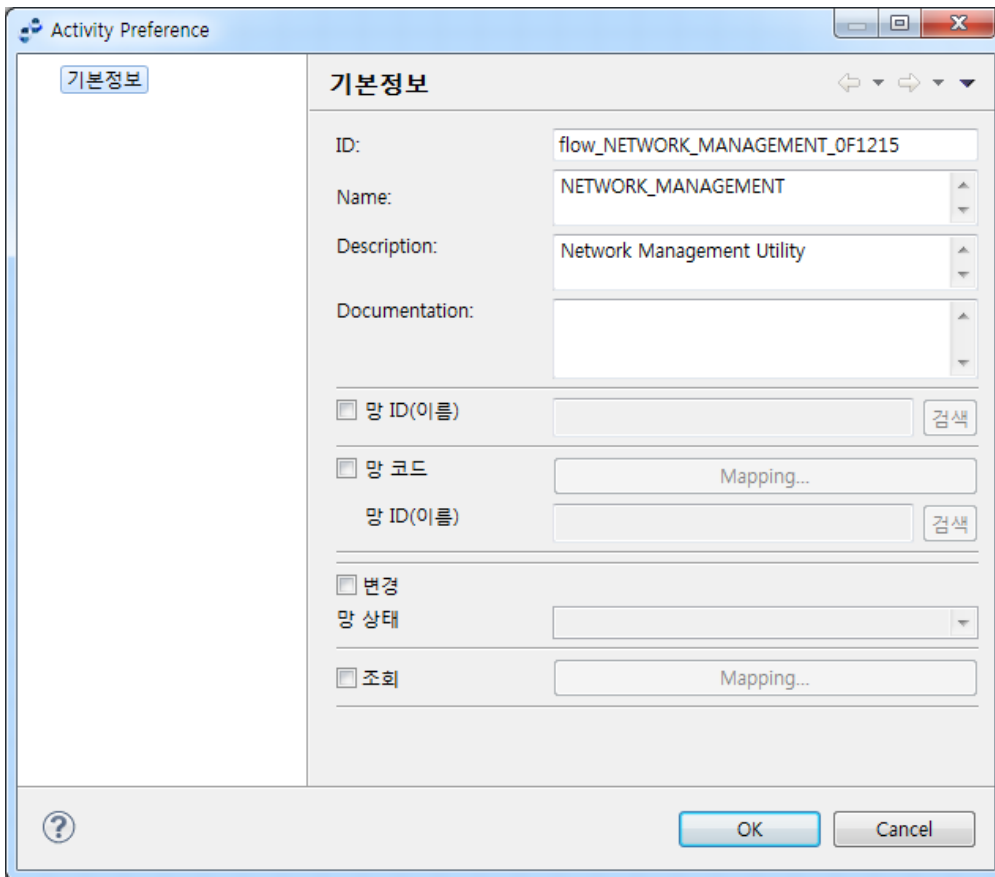
서비스 플로우에 추가되어 있는 '**Session ID**'를 선택한다. [Search] 버튼을 클릭하면 **Session ID 선택 다이얼로그**가 생성된다.



Session ID 선택 다이얼로그

6.8.2. NETWORK MANAGEMENT ACTIVITY

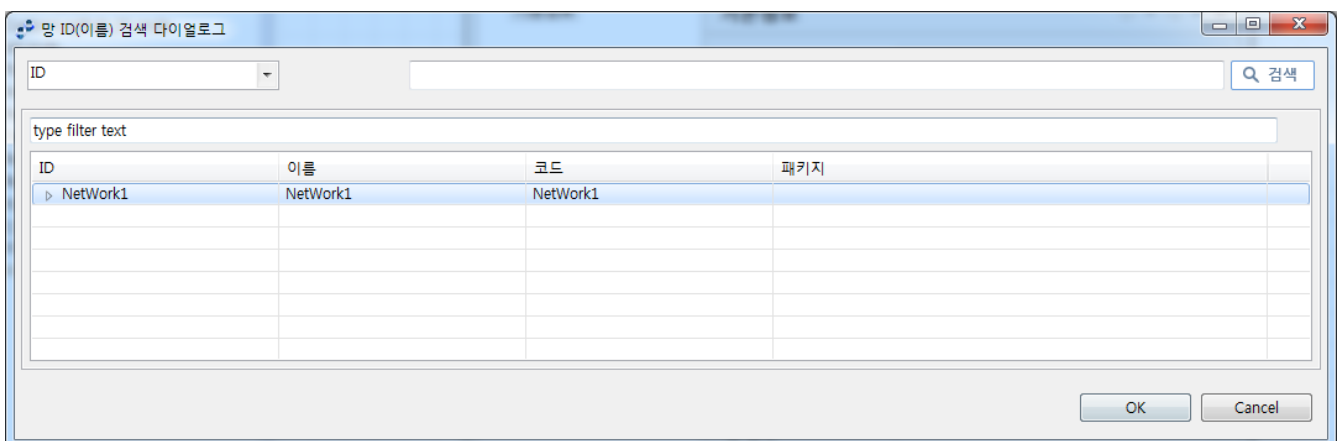
다음은 NETWORK MANAGEMENT ACTIVITY의 **Network Management Preference** 화면에 대한 설명이다.



Network Management Preference 화면

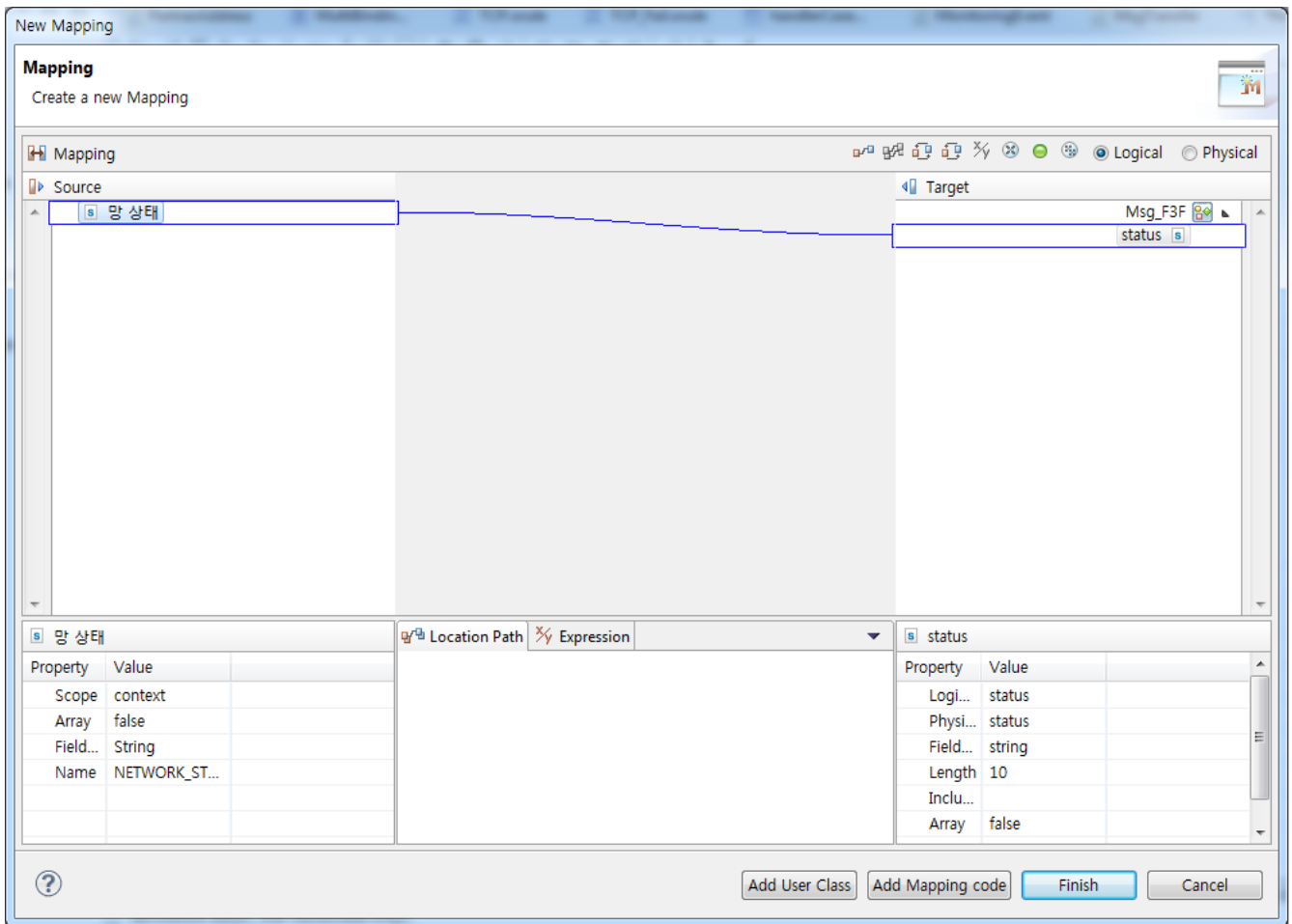
고정된 망 망을 선택할 경우에는 '망 ID(이름)'을 체크하고 [검색] 버튼을 클릭하여 생성된 망 ID(이름) 검색 다이얼로그에서 WebAdmin에서 추가한 망 ID를 선택한다.

플로우 변수에 코드 값이 있을 경우에는 '망 코드'을 체크하고 [Mapping...] 버튼을 클릭하여 생성된 코드를 매핑한 후 [검색] 버튼을 클릭하여 생성된 망 ID(이름) 검색 다이얼로그에서 WebAdmin에서 추가한 망 ID를 선택한다.



망 ID(이름) 검색 다이얼로그

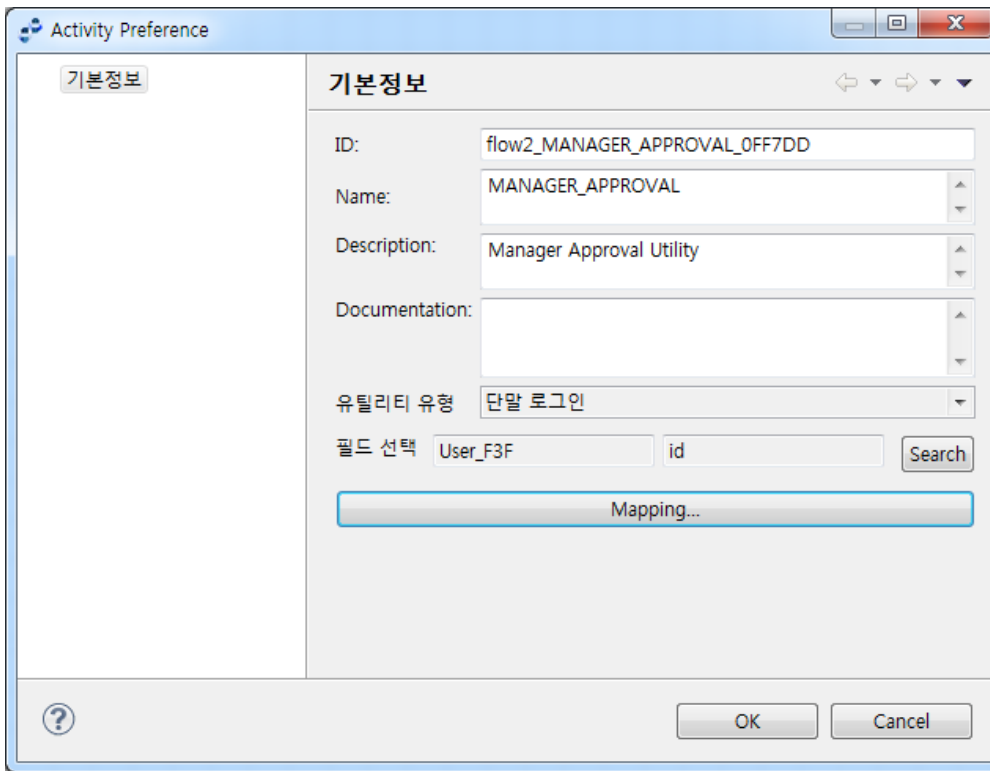
망 상태를 변경할 때는 '변경' 항목을 체크하고 망 상태를 변경한다. 망 상태를 조회할 때는 '조회' 항목을 체크하고 [Mapping...] 버튼을 클릭하여 생성된 망 상태 매핑 다이얼로그에서 망 상태를 플로우 변수에 매핑한다.



망 상태 매핑 다이얼로그

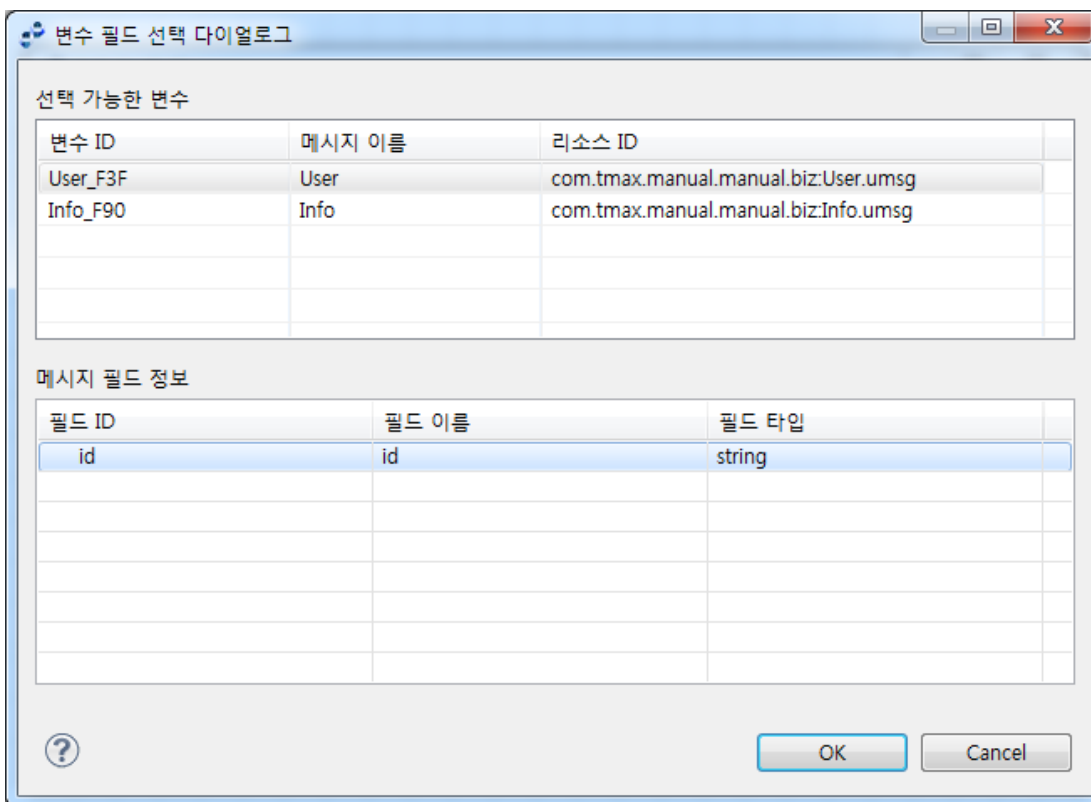
6.8.3. MANAGER APPROVAL ACTIVITY

다음은 MANAGER APPROVAL ACTIVITY의 **Manager Approval Preference 화면**에 대한 설명이다.



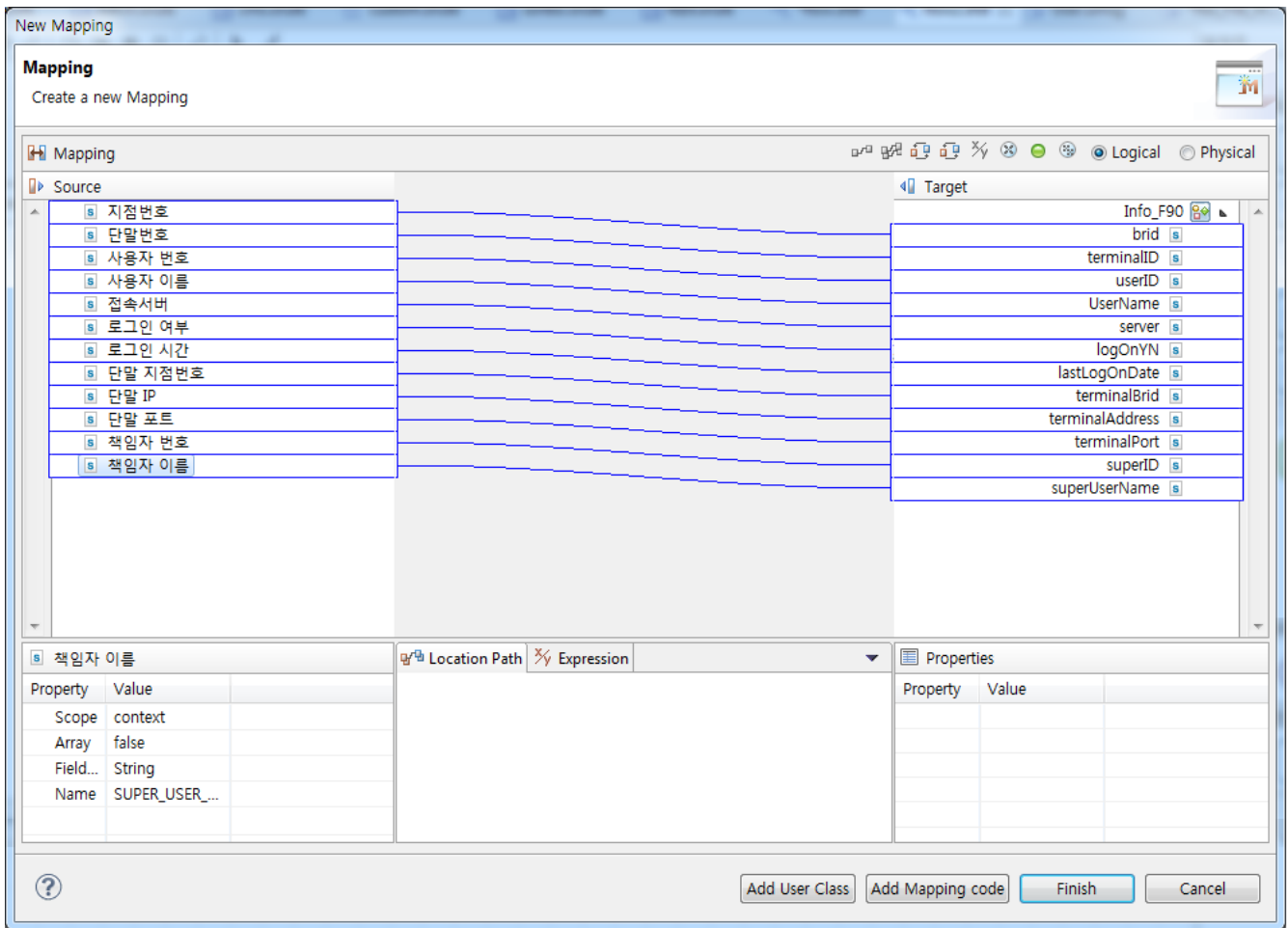
Manager Approval Preference 화면

유틸리티 유형에서 수행하려는 액션을 선택한다. 단말을 사용하거나 조회하려는 사용자의 사용자 번호를 **필드 선택**의 **[Search]** 버튼을 클릭하여 생성된 **변수 필드 선택 다이얼로그**에서 매핑한다. 해당 필드 값에 해당하는 값은 WebAdmin의 **[운영관리] > [단말관리] > [사용자관리]** 메뉴에 존재하는 사용자의 사용자 번호와 일치해야 한다.



변수 필드 선택 다이얼로그

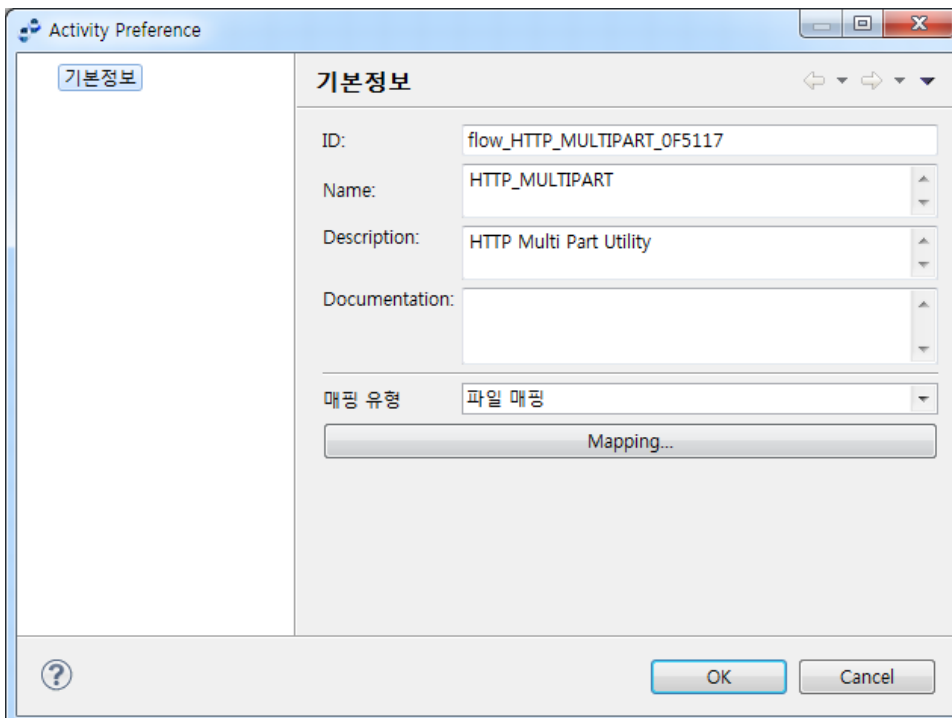
유틸리티 유형에 해당하는 액션을 수행한 후의 결과나 정보는 **[Mapping...]** 버튼을 클릭하여 생성된 **단말/사용자 정보 매핑 다이얼로그**에서 매핑한다.



단말/사용자 정보 매핑 다이얼로그

6.8.4. HTTP MULTIPART ACTIVITY

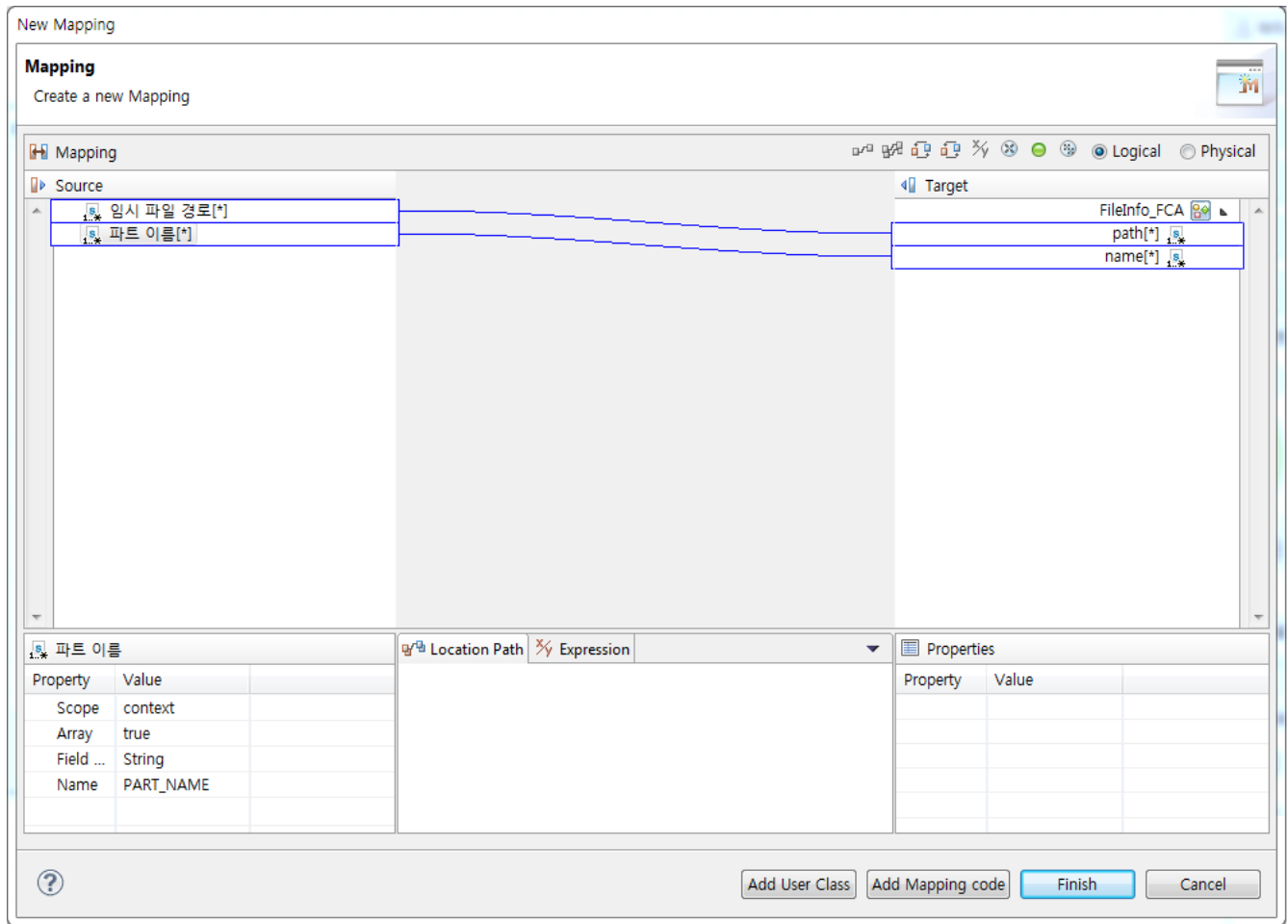
다음은 HTTP MULTIPART ACTIVITY의 **HTTP Multipart Preference 화면**에 대한 설명이다.



HTTP Multipart Preference 화면

멀티파트로 수신된 파일의 정보는 **[매핑 유형]**에서 유형을 선택한다. 파일 매핑은 파일의 경로와 파일 이름을 매핑을 통해 얻을 수 있고, 바이트 매핑은 파일의 내용과 파일의 정보를 매핑을 통해 얻을 수 있다.

멀티파트 파일 정보는 **[Mapping...]** 버튼을 클릭하여 생성된 **멀티 파트 매핑 다이얼로그**를 통해 매핑한다.

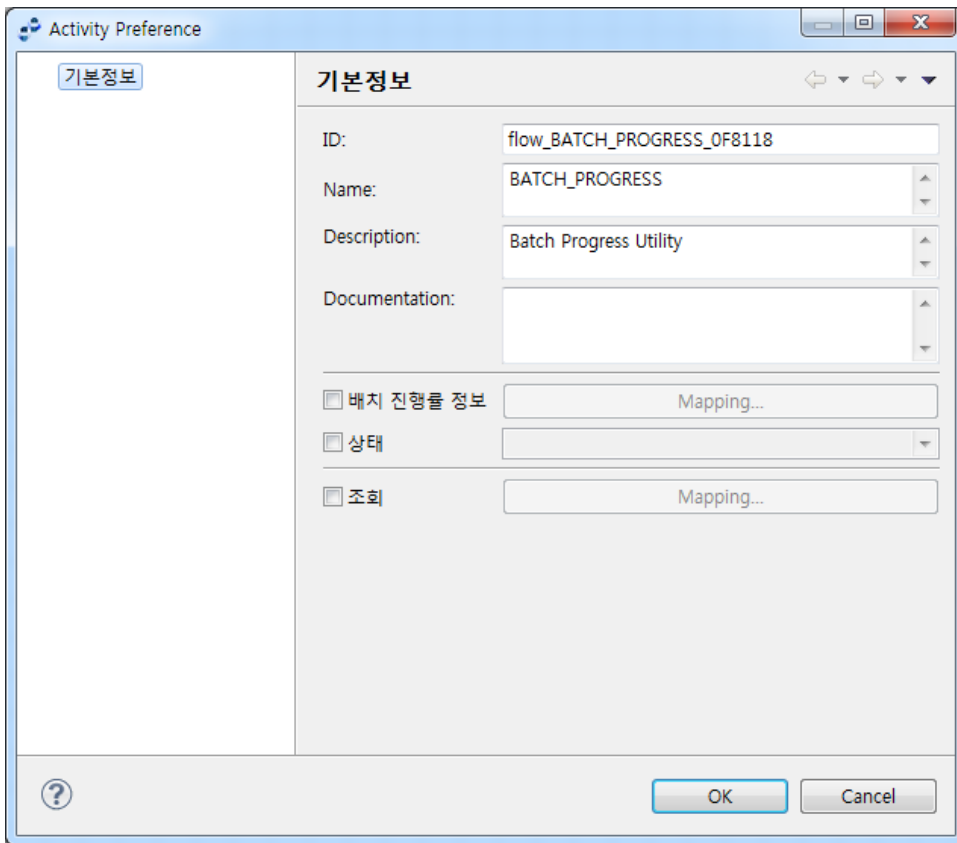


멀티 파트 매핑 다이얼로그

6.8.5. BATCH PROGRESS ACTIVITY

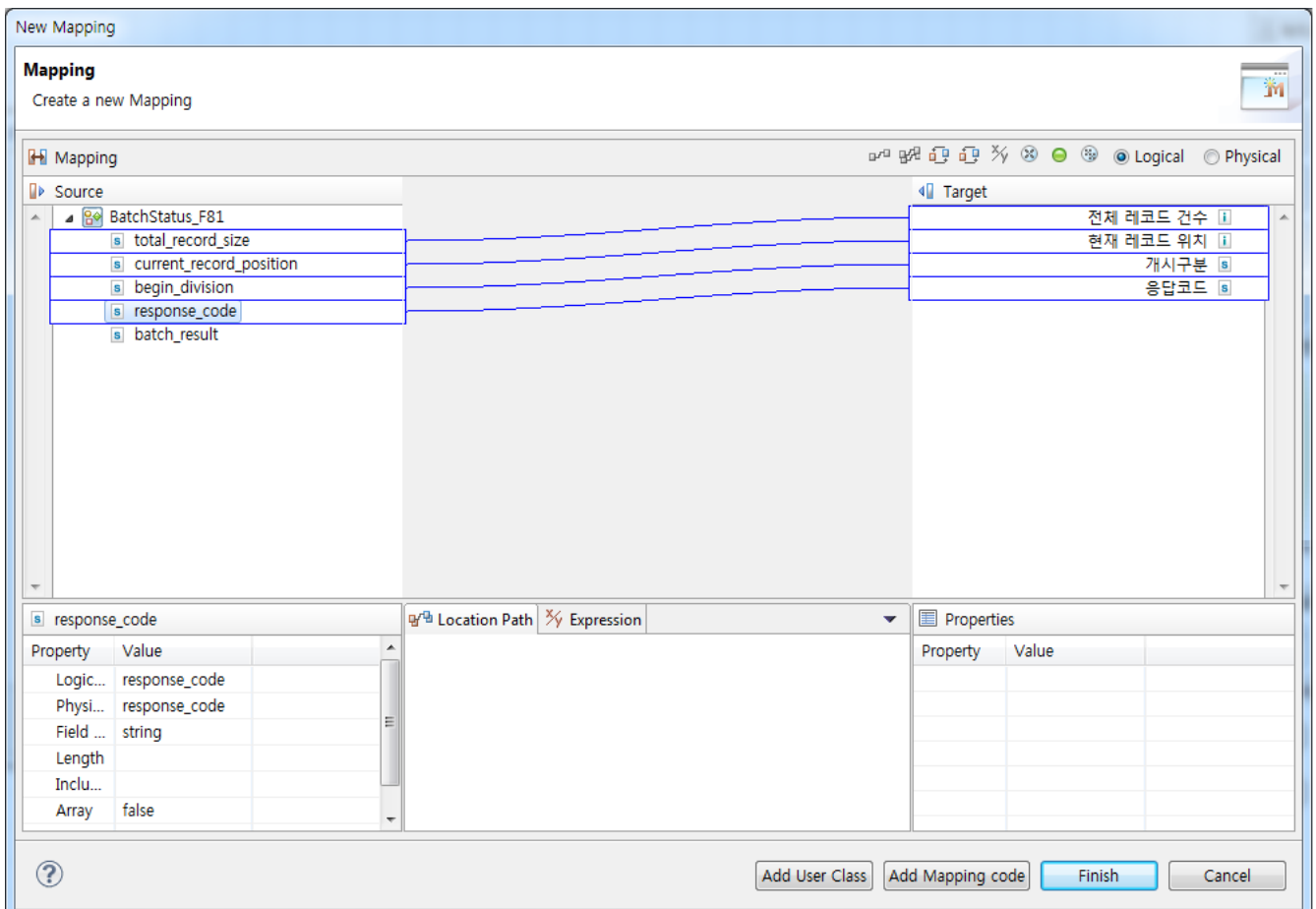
다음은 BATCH PROGRESS ACTIVITY의 **Batch Progress Preference 화면**에 대한 설명이다.

배치 진행률을 사용하기 위해서는 플로우에서 사용하는 배치 아웃바운드를 설정에서 배치 진행률 사용을 **true**로 설정하여야 한다.



Batch Progress Preference 화면

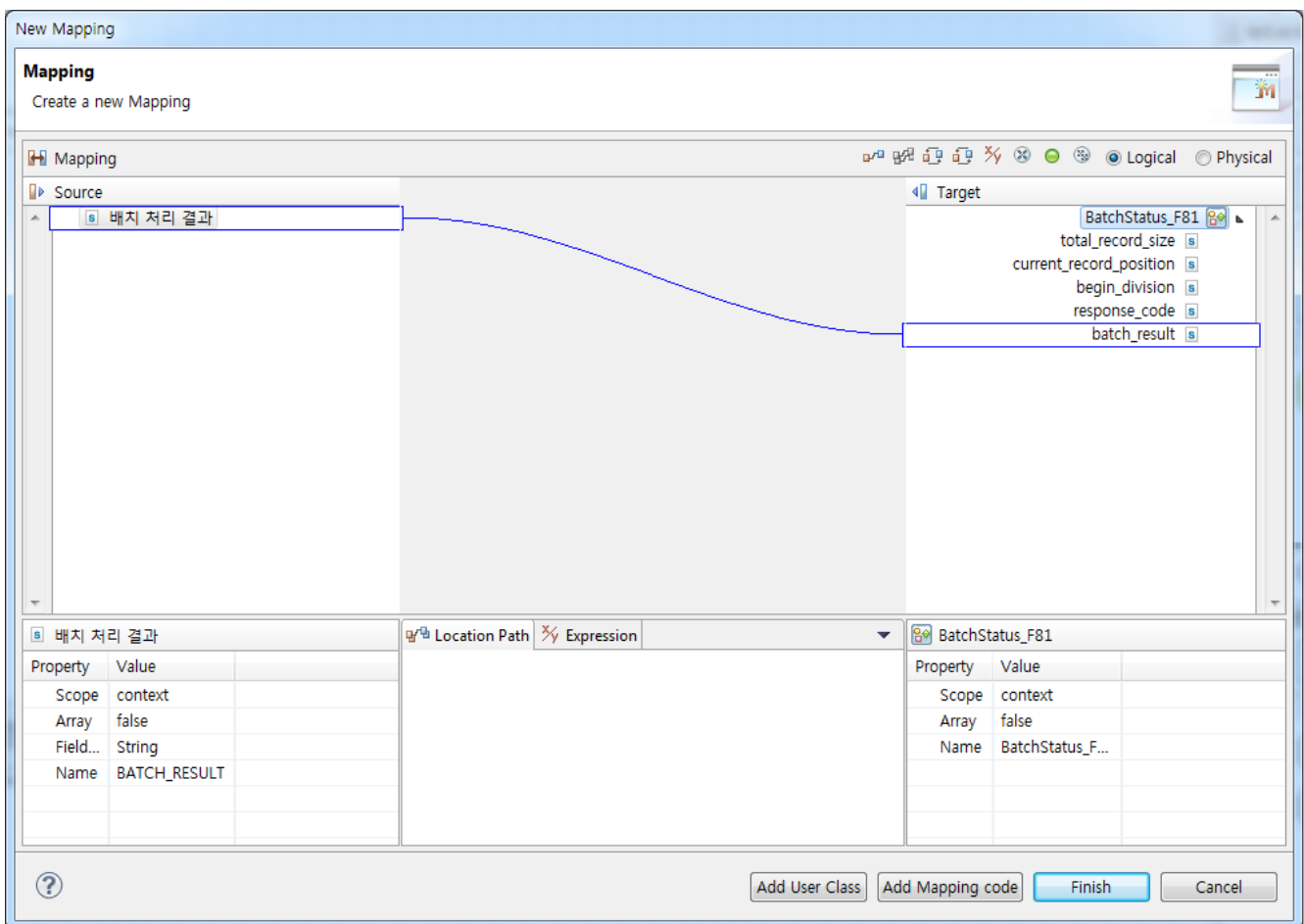
배치 진행률 정보를 변경할 경우에는 '배치 진행률 정보'를 체크하고 [Mapping...] 버튼을 클릭하여 생성된 배치 진행률 정보 매핑 다이얼로그에서 배치 진행률 정보를 매핑한다.



배치 진행률 정보 매핑 다이얼로그

항목	설명
배치 진행률 정보	배치 진행률 정보를 매핑하여 변경한다. <ul style="list-style-type: none"> ◦ 전체 레코드 건수 : 전체 레코드 건수를 변경한다. ◦ 현재 레코드 위치 : 현재 레코드 위치를 변경한다. ◦ 개시구분 : 개시 구분을 변경한다. ◦ 응답코드 : 응답코드를 변경한다.
상태	배치의 진행 상태를 완료 또는 실패로 변경한다.
조회	배치 처리 결과를 매핑을 통해 조회한다.

배치 상태를 변경할 경우에는 '상태'를 체크하고 변경할 상태를 선택한다. 배치 처리 결과를 조회할 경우에는 '조회'를 체크하고 **[Mapping...]** 버튼을 클릭하여 생성된 **배치 처리 결과 조회 매핑 다이얼로그**에서 배치 처리 결과를 매핑한다.

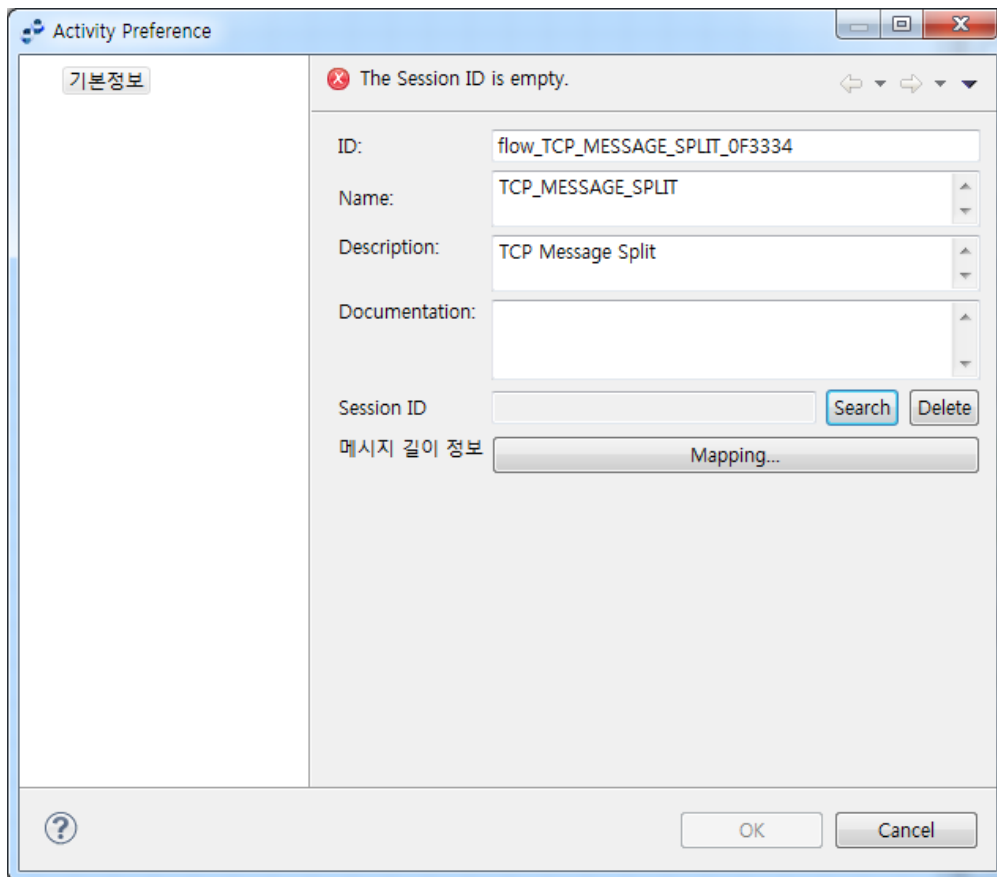


배치 처리 결과 조회 매핑 다이얼로그

6.8.6. TCP MESSAGE SPLIT ACTIVITY

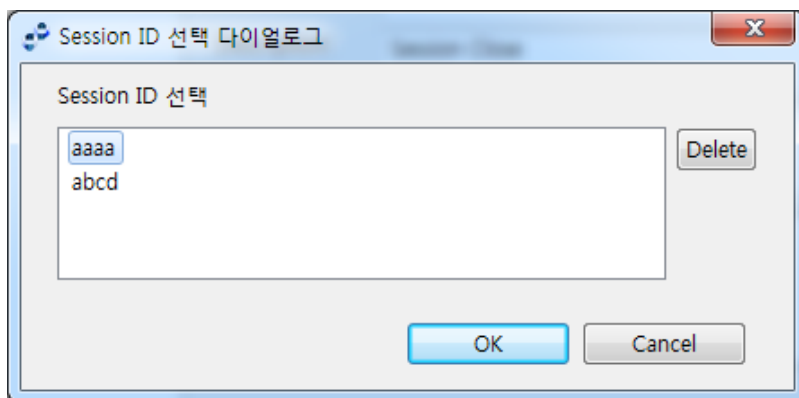
다음은 TCP MESSAGE SPLIT ACTIVITY의 **TCP Message Split Preference 화면**에 대한 설명이다. 해당 기능을 사용하기 위해서는 WebAdmin에 **[구성관리] > [어댑터]** 메뉴에서 Endpoint 설정에 메시지 분할 거래에 거래를 설정해야 한다.

TCP MESSAGE SPLIT ACTIVITY를 설정하면, 해당 Activity 다음 실행되는 Intermediate Event에서 메시지를 수신할 때 TCP MESSAGE SPLIT ACTIVITY 에서 설정한 '총 길이', '분할 길이'를 이용하여 수신한다. '총 길이'는 수신할 메시지의 총 길이이고, '분할 길이'는 분할해서 읽을 메시지 길이이다.



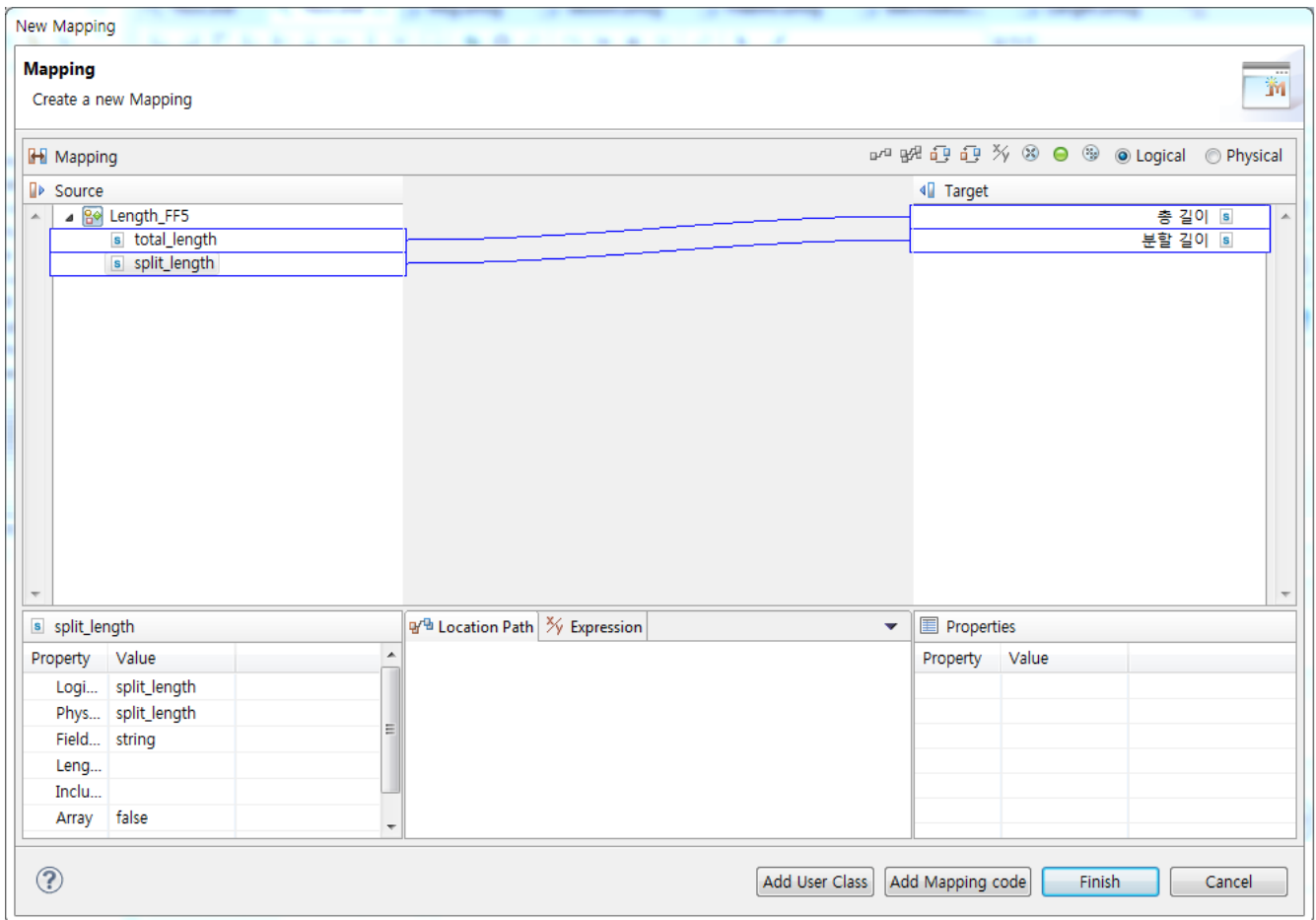
TCP Message Split Preference 화면

서비스 플로우에 추가되어 있는 '**Session ID**'를 선택한다. **[Search]** 버튼을 클릭하면 **Session ID 선택 다이얼로그**가 생성된다.



Session ID 선택 다이얼로그

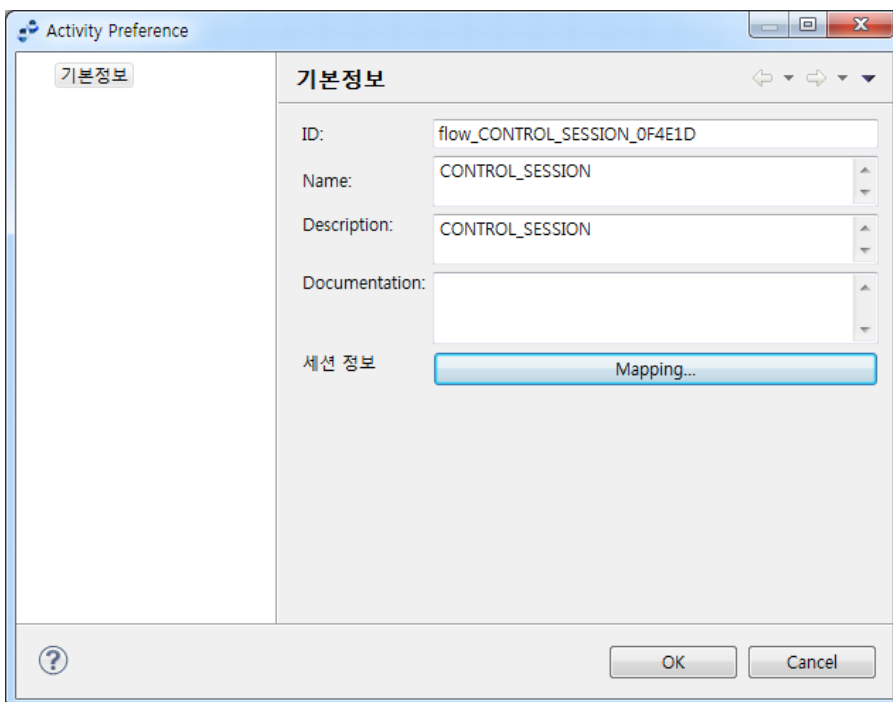
수신받을 메시지 길이 정보는 **[Mapping...]** 버튼을 클릭하여 생성된 **메시지 길이 정보 매핑 다이얼로그**에서 매핑한다. 이후에 수행되는 **Intermediate Event**는 **메시지 길이 정보 매핑 다이얼로그**에서 매핑된 총 길이와 분할 길이를 이용하여 수신한다.



메시지 길이 정보 매핑 다이얼로그

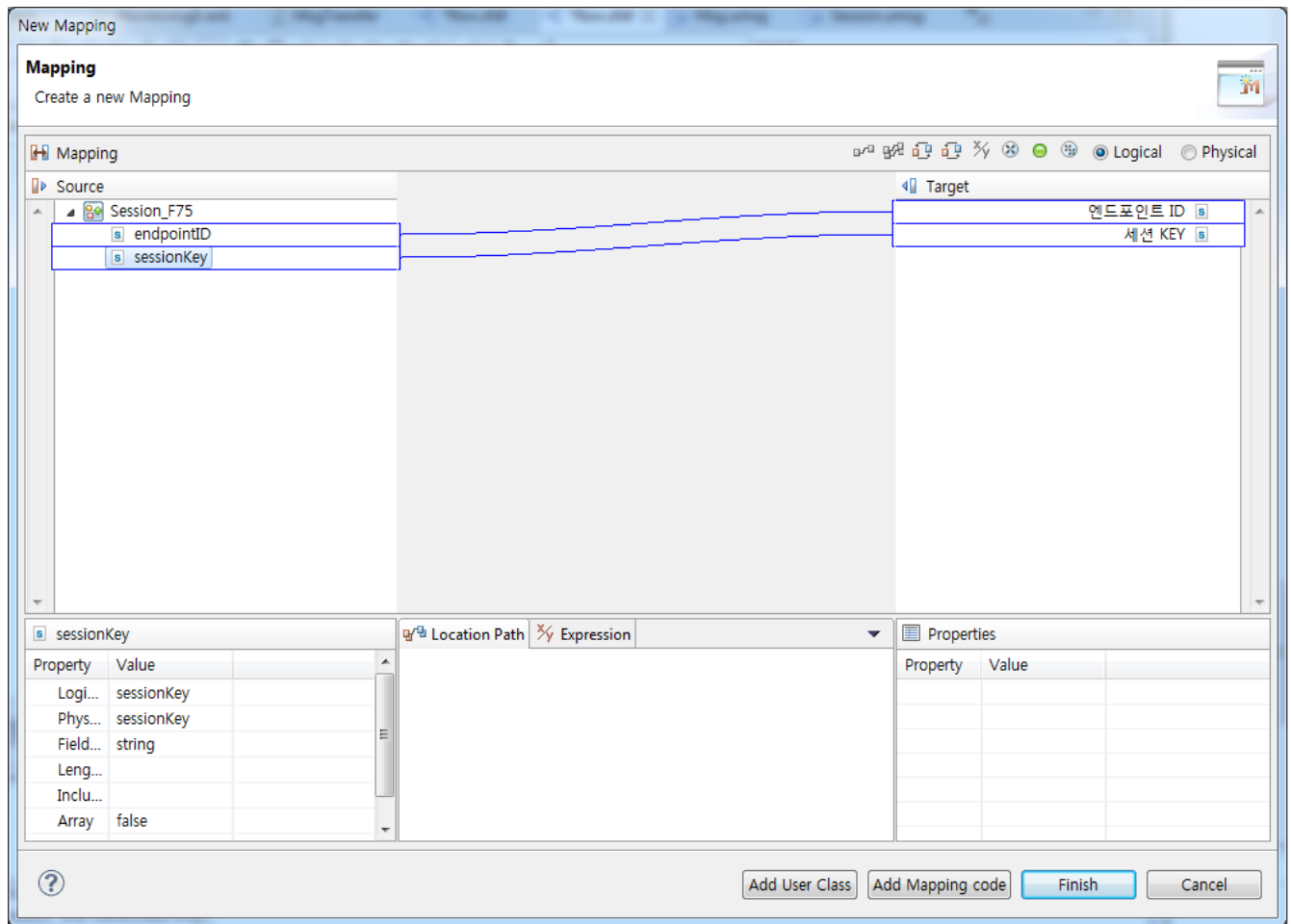
6.8.7. CONTROL SESSION ACTIVITY

다음은 CONTROL SESSION ACTIVITY의 **Control Session Preference** 화면에 대한 설명이다.



Control Session Preference 화면

세션을 종료하고 싶은 세션 정보는 **[Mapping...]** 버튼을 클릭하여 생성된 다이얼로그에서 매핑한다.



세션 정보 매핑 다이얼로그

7. 유저 클래스/핸들러 에디터

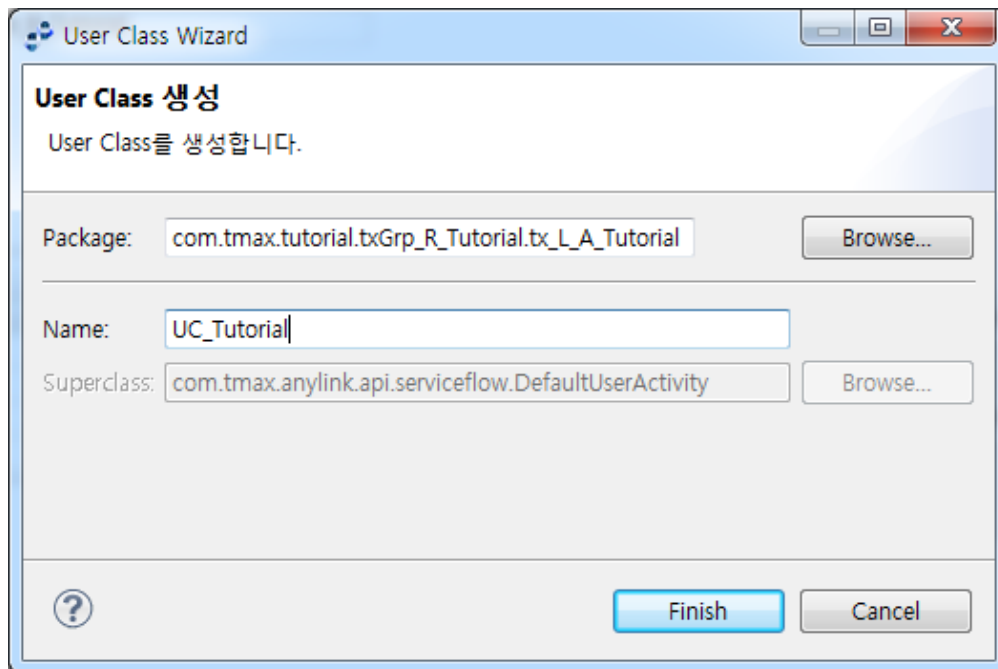
본 장에서는 AnyLink 스튜디오의 유저 클래스/핸들러 에디터의 기능과 사용법에 대해 설명한다.

7.1. 유저 클래스

유저 클래스는 플로우 조건의 표현식으로 정의할 수 있는 로직보다 복잡한 로직을 정의해야 하는 경우에 사용된다.

7.1.1. 유저 클래스 생성

유저 클래스의 생성은 왼쪽 거래 네비게이터의 컨텍스트 메뉴에서 **[새로만들기] > [유저 클래스]**를 선택한다. **유저 클래스 생성 화면**에서 각 항목을 입력하고 **[Finish]** 버튼을 클릭한다.

The image shows a 'User Class Wizard' dialog box. The title bar says 'User Class Wizard'. Inside, the text 'User Class 생성' (User Class Creation) is followed by 'User Class를 생성합니다.' (Create User Class). There are three input fields: 'Package:' with the value 'com.tmax.tutorial.txGrp_R_Tutorial.tx_L_A_Tutorial', 'Name:' with the value 'UC_Tutorial', and 'Superclass:' with the value 'com.tmax.anylink.api.serviceflow.DefaultUserActivity'. Each field has a 'Browse...' button to its right. At the bottom, there is a question mark icon on the left, and 'Finish' and 'Cancel' buttons on the right.

유저 클래스 생성 화면

항목	설명
Package	유저 클래스를 생성할 거래를 선택한다.
Name	유저 클래스의 이름을 입력한다. Java의 클래스명 생성법과 동일하다.
Superclass	유저 클래스가 상속 받게 될 클래스이다. DefaultUserActivity 클래스가 기본으로 생성된다.

7.1.2. 유저 클래스 정의

유저 클래스가 생성되면 에디터 화면으로 이동해서 유저 클래스를 작성할 수 있다.

유저 클래스는 Java 소스와 동일한 형태이며 DefaultUserActivity 클래스를 상속 받은 클래스이다. action 메소드가 override되어 생성된다.

```

package com.tmax.tc_tutorial.txGrp_R_tutorial.tx_L_A_tutorial;

import com.tmax.anylink.api.serviceflow.ActivityContext;
import com.tmax.anylink.api.serviceflow.DefaultUserActivity;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class UC_Tutorial extends DefaultUserActivity {

    private static final Logger logger = Logger.getLogger(UC_Tutorial.class.getName());

    public void action(ActivityContext arg0) throws AnyLinkException {
        // TODO Auto-generated method stub

    }

}

```

7.1.3. getEnv 함수

getEnv 함수는 유저 클래스나 핸들러에서 시스템 정보나 거래의 정보들을 가져올 때 사용할 수 있다.

다음은 getEnv 함수에서 사용 가능한 key 값과 해당 key를 이용해 가져올 수 있는 값에 대한 설명이다.

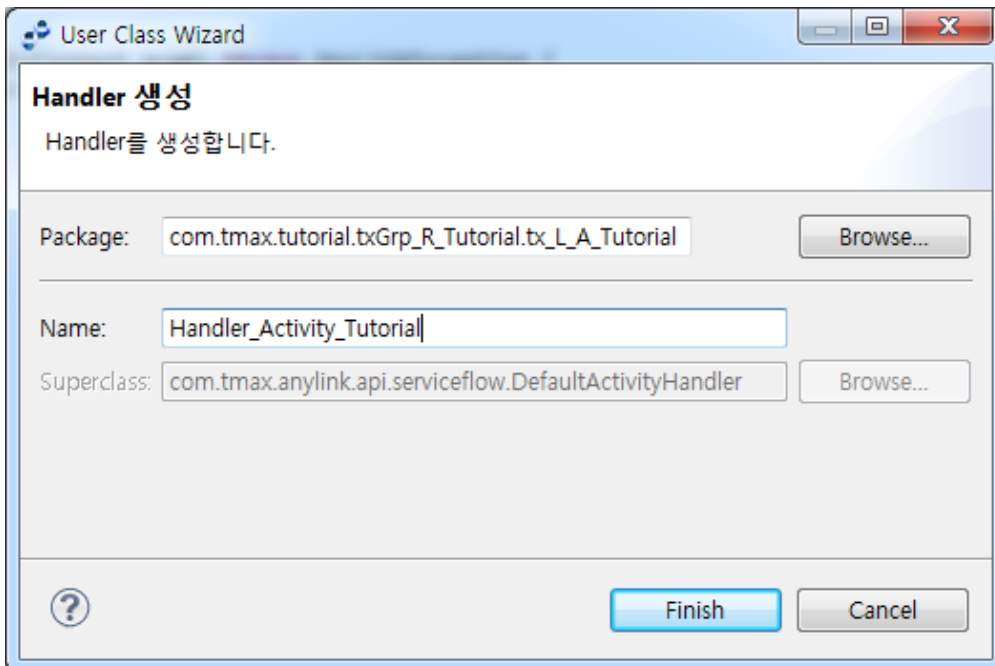
항목	설명
install.root	AnyLink가 설치된 루트 경로를 가져온다.
domain.home	도메인의 home 경로를 가져온다.
server.home	서버의 home 경로를 가져온다.
server.name	서버의 이름을 가져온다.
adminServer.name	adminServer의 이름을 가져온다.
cluster.name	클러스터의 이름을 가져온다.
bizsystem.id	업무시스템의 ID를 가져온다.
hostname	hostname을 가져온다.
biztx.id	거래 ID를 가져온다.
guid	GUID를 가져온다.
startendpoint.id	거래가 시작된 엔드포인트 ID를 가져온다.
startadapter.id	거래가 시작된 어댑터 ID를 가져온다.
startendpoint.sysid	거래가 시작된 엔드포인트의 SysID를 가져온다.
startendpoint.conne ction.count	거래가 시작된 엔드포인트의 커넥션 개수를 가져온다.
request.ip	거래를 요청한 상대의 IP를 가져온다. 잡 스케줄로 실행하는 경우 잡의 아이디를 가져온다.
request.port	거래를 요청한 상대의 port를 가져온다.

항목	설명
request.url	거래를 요청한 url을 가져온다.
request.http.cookie	HTTP 요청으로 실행된 거래에 대해 HTTP Request에 설정된 쿠키를 가져온다.
connection.guid	커넥션 GUID를 가져온다.
biztx.name	거래 이름을 가져온다.
biztx.code	거래 코드를 가져온다.
biztx.type	거래 타입을 가져온다.
correlation.value	코릴레이션 값을 가져온다. 매핑 핸들러 또는 매핑 액티비티에서 사용가능하다.
local.ip	local IP를 가져온다.
local.port	local Port를 가져온다.
connect.type	연결 타입을 가져온다.
hostname	hostname을 가져온다.
\${sys:변수명}	WebAdmin에서 설정한 시스템 변수를 얻는다.

7.2. 핸들러

7.2.1. 핸들러 생성

유저 클래스의 생성은 왼쪽 거래 네비게이터의 컨텍스트 메뉴에서 **[새로만들기] > [핸들러] > [생성할 핸들러 종류]**를 선택한다. **핸들러 생성 화면**에서 각 항목을 입력하고 **[Finish]** 버튼을 클릭한다.



핸들러 생성 화면(액티비티 핸들러)

항목	설명
Package	핸들러를 생성할 거래를 선택한다.
Name	핸들러의 이름을 입력한다. Java의 클래스명 생성법과 동일하다.
Superclass	각 핸들러가 상속받게 될 클래스를 보여준다.

7.2.2. 핸들러 정의

핸들러가 생성되면 에디터 화면으로 이동해서 핸들러를 작성할 수 있다. 해당 핸들러는 Java 소스와 동일한 형태이며 각 핸들러의 슈퍼 클래스를 상속 받은 클래스이다.

다음은 AnyLink에서 제공되는 핸들러의 목록이다.

구분	Super Class	설명
액티비티	DefaultActivityHandler	액티비티의 시작/종료 등 특정 시점에 수행할 동작을 정의한다.
액티비티 에러	DefaultActivityErrorHandler	액티비티에 에러가 발생하는 경우 수행할 동작을 정의한다.
프로세스	DefaultProcessHandler	프로세스의 시작/종료 등 특정 시점에 수행할 동작을 정의한다.
프로세스 에러	DefaultProcessErrorHandler	프로세스에 에러가 발생하는 경우 수행할 동작을 정의한다.
어댑터 메시지	DefaultAdapterMessageHandler	메시지를 엔드포인트에 전달하거나 전달받기 전에 메시지에 대한 전처리를 한다.
암복호화	DefaultEncryptionHandler	로깅할 때 사용자가 원하는 암복호화를 정의한다.
유저 매핑	DefaultUserMappingHandler	매핑 다이얼로그에서 지원하지 않는 복잡한 매핑을 한다.
파싱	DefaultParsingHandler	사용자가 원하는 메시지를 파싱한다.
타이머	DefaultUserTimer	타이머 이벤트에서 추가 설정을 한다.
메시지 유효성 검사	DefaultMessageValidityHandler	생성한 메시지의 유효성을 확인하기 위한 로직을 작성한다.
라우팅 핸들러	DefaultRoutingHandler	멀티바인딩에서 바인딩 설정의 로직을 작성한다.
SSO 로그인 핸들러	DefaultSsoLoginHandler	SSO 로그인을 위한 로직을 작성한다.
모니터링 이벤트 핸들러	DefaultMonitoringEventHandler	모니터링 중에 이벤트 발생시 해당 이벤트 처리에 대한 로직을 작성한다.
전문 관리 핸들러	DefaultMsgTransferHandler	전문 공유를 위한 로직을 작성한다.
기관 담당자 핸들러	DefaultPartnerAddressHandler	대외 연락처에 설정된 담당자 정보를 이용해 호출할 로직을 작성한다.
코릴레이션 핸들러	DefaultCorrelationHandler	요청/응답에 있는 코릴레이션 값 처리에 대한 로직을 작성한다.

액티비티 핸들러

액티비티가 시작, 종료, 취소, 에러 종료 시점에 호출되는 핸들러이다.

핸들러	설명
started	액티비티가 시작되는 시점에 호출된다.
finished	액티비티가 완료되는 시점에 호출된다.
cancelled	액티비티의 취소 완료 시점에 호출된다.
errorOccurred	액티비티의 에러 완료 시점에 호출된다.

다음은 액티비티의 시작, 종료, 취소, 에러 종료 시점에 액티비티 아이디를 로그로 남기는 예제 핸들러이다.

```
package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.serviceflow.ActivityContext;
import com.tmax.anylink.api.serviceflow.DefaultActivityHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class ActivityHandlerImpl extends DefaultActivityHandler {

    private static Logger logger = Logger.getLogger(ActivityHandlerImpl.class.getName());

    @Override
    public void started(ActivityContext paramActivityContext)
        throws AnyLinkException {
        // TODO Auto-generated method stub
        logger.info("==== Activity Started : " + paramActivityContext.getActivityId() );
    }

    @Override
    public void finished(ActivityContext paramActivityContext)
        throws AnyLinkException {
        // TODO Auto-generated method stub
        logger.info("==== Activity Finished : " + paramActivityContext.getActivityId() );
    }

    @Override
    public void cancelled(ActivityContext paramActivityContext,
        String paramString) throws AnyLinkException {
        // TODO Auto-generated method stub
        logger.info("==== Activity Canceled : " + paramActivityContext.getActivityId() );
    }

    @Override
    public void errorOccurred(ActivityContext paramActivityContext,
        Throwable paramThrowable) throws AnyLinkException {
        // TODO Auto-generated method stub
        logger.info("==== Activity Error Occured : " + paramActivityContext.getActivityId() );
    }
}
```

핸들러 작성이 완료되면 플로우 액티비티의 **[Activity Preference]**의 **액티비티 핸들러 설정 화면**에서 **[Add...]** 버튼을 클릭해서 핸들러를 등록할 수 있다.

액티비티 에러 핸들러

액티비티 실행 도중 특정 에러가 발생하는 경우 호출되는 핸들러이다.

다음은 액티비티 실행 중 에러가 발생하는 경우 액티비티 아이디와 에러 메시지를 로그로 남기는 예제 핸들러이다.

```
package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.serviceflow.ActivityContext;
import com.tmax.anylink.api.serviceflow.DefaultActivityErrorHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class ActivityErrorHandlerImpl extends DefaultActivityErrorHandler {

    private static Logger logger = Logger.getLogger(ActivityErrorHandlerImpl.class.getName());
    @Override
    public void handle(ActivityContext paramActivityContext,
        Throwable paramThrowable) throws AnyLinkException {
        // TODO Auto-generated method stub
        logger.info("==== Activity ID : "+paramActivityContext.getActivityId());
        logger.info("==== Error Message : "+paramThrowable.getMessage());
    }
}
```

핸들러 작성이 완료되면 플로우 액티비티의 **[Activity Preference]**의 **Activity Error Handler 설정 화면**에서 **[Add...]** 버튼을 클릭해서 에러 코드와 함께 핸들러를 등록할 수 있다. 등록된 핸들러는 함께 등록된 에러 코드를 갖는 에러가 발생하는 경우 호출된다.

프로세스 핸들러

서비스 플로우 시작, 종료 시점에 호출되는 핸들러이다.

다음은 플로우 시작, 종료 시점에 플로우 아이디를 로그로 남기는 핸들러 예제이다.

```
package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.serviceflow.DefaultProcessHandler;
import com.tmax.anylink.api.serviceflow.ProcessContext;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class ProcessHandlerImpl extends DefaultProcessHandler {

    private Logger logger = Logger.getLogger(ProcessHandlerImpl.class.getName());
    @Override
    public void finished(ProcessContext ctx) throws AnyLinkException {
        // TODO Auto-generated method stub
    }
}
```

```

        super.finished(ctx);
        logger.info("=== Flow Finished : " + ctx.getProcessId());
    }

    @Override
    public void started(ProcessContext ctx) throws AnyLinkException {
        // TODO Auto-generated method stub
        super.started(ctx);
        logger.info("=== Flow Started : " + ctx.getProcessId());
    }
}

```

핸들러 작성이 완료되면 플로우 에디터 컨텍스트 메뉴에서 **[Property]**를 선택하고 **[프로세스 핸들러]** 탭의 **[Add...]** 버튼을 클릭해서 플로우에 등록하여 사용한다.

프로세스 에러 핸들러

플로우 실행 도중 특정 에러가 발생하는 경우 호출되는 핸들러이다.

다음은 플로우 실행 중 에러가 발생하는 경우 플로우 아이디와 에러 메시지를 로그로 남기는 예제 핸들러이다.

```

package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.serviceflow.DefaultProcessErrorHandler;
import com.tmax.anylink.api.serviceflow.ProcessContext;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class ProcessErrorHandlerImpl extends DefaultProcessErrorHandler {

    private static Logger logger = Logger.getLogger(ProcessErrorHandlerImpl.class.getName());
    @Override
    public void handle(ProcessContext context, Throwable error)
        throws AnyLinkException {
        // TODO Auto-generated method stub
        logger.info("=== Flow ID : " + context.getProcessId());
        logger.info("=== Error Message : " + error.getMessage());
    }
}

```

핸들러 작성이 완료되면 플로우 에디터 컨텍스트 메뉴에서 **[Property]**를 선택하고 **[에러 핸들러]** 탭의 **[Add...]** 버튼을 클릭해서 에러 코드와 함께 핸들러를 등록할 수 있다. 등록된 핸들러는 함께 등록된 에러 코드를 갖는 에러가 발생하는 경우 호출된다.

어댑터 메시지 핸들러

엔드포인트를 통해 입출력되는 메시지를 변환하기 위한 핸들러이다. **receive** 메소드를 통해 입력된 메시지를 변환하여 반환하고 **send** 메소드를 통해 출력할 메시지를 변환하여 반환한다.

다음은 입력된 메시지를 소문자로 변경하고, 출력할 메시지를 대문자로 변경하는 핸들러 예제이다.

```

package com.tmax.pkg.adapterMSGCase1.tx1;

import com.tmax.anylink.api.adapter.DefaultAdapterMessageHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class AdapterMessageHandlerImpl extends DefaultAdapterMessageHandler {

    private static Logger logger = Logger.getLogger(AdapterMessageHandlerImpl.class.getName());

    @Override
    public Object receive(Object arg0) throws AnyLinkException {
        // TODO Auto-generated method stub
        String receiveMsg = new String((byte[]) arg0);
        logger.info("==== RECEIVE MSG : " + receiveMsg );

        return receiveMsg.toLowerCase().getBytes();
    }

    @Override
    public Object send(Object arg0) throws AnyLinkException {
        // TODO Auto-generated method stub
        String sendMsg = new String((byte[]) arg0);
        logger.info("==== SEND MSG : " + sendMsg );

        return sendMsg.toUpperCase().getBytes();
    }
}

```

핸들러 작성이 완료되면 핸들러가 포함된 거래를 배포함으로써 사용이 가능하다. 거래가 배포되고 난 후 WebAdmin의 **엔드포인트 설정 화면**에서 **[상세설정]** 탭에 **'메시지 핸들러'**에 등록하여 사용한다.

암복호화 핸들러

트레이스 로그, 커스텀 로그를 저장, 조회하는 경우 메시지의 암복호화를 처리하기 위한 핸들러이다. **decrypt** 메소드를 통해 메시지를 복호화 처리를 하여 반환하고 **encrypt** 메소드를 통해 메시지를 암호화 처리하여 반환한다.

다음은 BASE64 인코딩 방식으로 암복호화 처리를 하는 핸들러 예제이다.

```

package com.tmax.pkg.encryptionCase1.tx1;

import com.tmax.anylink.api.log.DefaultEncryptionHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.common.util.Base64;
import com.tmax.anylink.logging.Logger;

public class EncryptionHandlerImpl extends DefaultEncryptionHandler {

    private static Logger logger = Logger.getLogger(EncryptionHandlerImpl.class.getName());

    @Override
    public byte[] decrypt(byte[] arg0) throws AnyLinkException {

```



```

// TODO Auto-generated method stub

logger.info("==== Encrypted MSG : " + new String(arg0) );

Base64 decoder = new Base64();
byte[] decryptedMessage = decoder.decode(new String(arg0));

return decryptedMessage;
}

@Override
public byte[] encrypt(byte[] arg0) throws AnyLinkException {
    // TODO Auto-generated method stub

    logger.info("==== Original MSG : " + new String(arg0) );

    Base64 eecoder = new Base64();
    byte[] encryptedMessage = encrypt(arg0);

    return encryptedMessage;
}
}

```

작성된 핸들러는 스튜디오 **커스텀 로그 아웃바운드 룰**의 **'암복호화 사용'**을 선택한 후 **'유저 클래스 선택'** 옵션에 등록하여 사용한다. 핸들러를 포함한 거래가 배포되고 난 후 WebAdmin의 로그 어댑터 **[트레이스 로그]** 탭의 **'암호화 User Class 이름'**에 패키지 명을 포함하여 입력하여 등록한 후 사용한다.

유저 매핑 핸들러

매핑 다이얼로그에 설정하여 매핑을 사용자가 직접 정의하는 경우 사용하는 핸들러이다. 매핑 전달인자 중 **arg1**이 매핑의 Source에 해당하고 반환값인 **Object[]**가 매핑된 Target에 해당한다.

다음은 Input, Output을 순차적으로 1:1 매핑하는 핸들러 예제이다.

```

package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.serviceflow.ActivityContext;
import com.tmax.anylink.api.serviceflow.DefaultUserMapping;
import com.tmax.anylink.common.AnyLinkException;

public class UserMappingHandlerImpl extends DefaultUserMapping {

    @Override
    public Object[] mapping(ActivityContext arg0, Object[] arg1)
        throws AnyLinkException {
        // TODO Auto-generated method stub
        Object[] mappedTarget = new Object[arg1.length];
        for(int i = 0; i < arg1.length; i++){
            mappedTarget[i] = arg1[i];
        }
        return mappedTarget;
    }
}

```

핸들러 작성이 완료되면 매핑 다이얼로그의 **[Add User Class]** 버튼을 클릭해서 핸들러를 등록할 수 있다.

파싱 핸들러

거래그룹 에디터의 파싱 정보에 '하위 거래 식별 방법' 항목이 'HANDLER'로 설정된 경우 사용되는 핸들러이다. 핸들러의 반환값에 해당하는 거래 식별 코드를 갖는 거래를 선택하여 파싱하기 위해 사용된다.

다음은 요청 메시지의 앞 4자리를 거래 식별 코드로 생성하는 핸들러 예제이다.

```
package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.adapter.DefaultParsingHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class ParsingHandlerImpl extends DefaultParsingHandler {

    private static Logger logger = Logger.getLogger(ParsingHandlerImpl.class.getName());

    @Override
    public String parsing(Object arg0) throws AnyLinkException {
        // TODO Auto-generated method stub
        String inputMsg = new String((byte[]) arg0);
        logger.info("====Input Message : " + inputMsg);
        String parsingMsg = inputMsg.substring(0, 3);
        logger.info("====Parsing Message : " + parsingMsg);

        return parsingMsg;
    }
}
```

핸들러 작성이 완료되면 거래그룹 에디터의 파싱 정보에 '하위 거래 식별 방법' 항목을 'HANDLER'로 선택한 후 핸들러를 등록할 수 있다.

타이머 핸들러

타이머 이벤트의 설정을 위한 핸들러이다. **nextTime** 메소드는 타이머 이벤트가 발생시킬 시각을 설정하여 반환하고 **shouldFireTimerEvent** 메소드는 타이머 이벤트를 발생시킬지 여부를 반환한다.

다음은 타이머 이벤트를 500ms에 무조건 타이머 이벤트를 발생시키는 핸들러 예제이다.

```
package com.tmax.pkg.handlerCase1;

import java.util.logging.Logger;

import com.tmax.anylink.api.serviceflow.ActivityContext;
import com.tmax.anylink.api.serviceflow.DefaultUserTimer;

public class TimerHandlerImpl extends DefaultUserTimer {
```

```

private static Logger logger = Logger.getLogger(TimerHandlerImpl.class.getName());
private long startTime = -1;
private final static long PERIOD = 20;

@Override
public long nextTime(ActivityContext arg0) {
    // TODO Auto-generated method stub
    if(startTime < 0) {
        startTime = System.currentTimeMillis() + PERIOD ;
    } else {
        startTime += PERIOD ;
    }

    logger.info("##### TimerHandlerImpl startTime : " + startTime);

    return startTime;
}

@Override
public boolean shouldFireTimerEvent(ActivityContext arg0, long arg1) {
    // TODO Auto-generated method stub
    return true;
}
}

```

핸들러 작성이 완료되면 **[Timer Event] > [Event Preference]** 화면의 **[Timer]** 탭에 **'Timer Class Name'** 항목에 핸들러를 입력하여 사용한다.

메시지 유효성 검사 핸들러

생성한 메시지의 유효성(메타 데이터 포함 여부)을 판단하기 위한 로직을 작성하는 핸들러이다.

다음은 메시지의 **Physical Name**에 'ab'가 포함되는 경우 유효하지 않은 메시지로 판단하는 핸들러 예제이다.

```

package com.tmax.pkg.msgValidityCase1;

import java.util.HashMap;
import java.util.Map;

import com.tmax.anylink.api.validation.DefaultMessageValidityHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmaxsoft.promapper.structure.StructureFieldType;

public class MsgValidityImpl extends DefaultMessageValidityHandler {

    @Override
    public Map<StructureFieldType, Boolean> messageMetaCheck(
        StructureFieldType[] arg0) throws AnyLinkException {
        // TODO Auto-generated method stub
        Map<StructureFieldType, Boolean> ret = new HashMap<StructureFieldType, Boolean>();

        for(StructureFieldType type : arg0){
            if(type.getPhysicalName().contains("ab")){ // 유효하지 않은 메시지 필드
                ret.put(type, true); // 해당 메시지 필드와 'true' 값을 맵에 저장
            }
        }
    }
}

```

```

    }else{ // 유효한 메시지 필드
        ret.put(type, false); // 해당 메시지 필드와 'false' 값을 맵에 저장
    }
}
return ret;
}
}

```

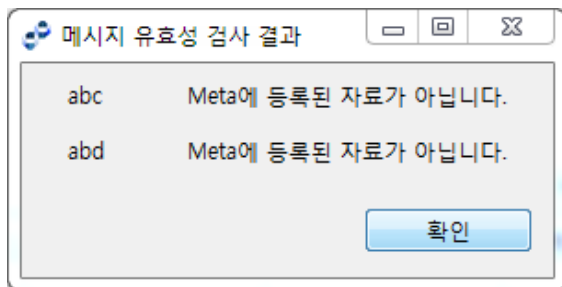
핸들러 작성이 완료되면 네비게이터에서 생성한 핸들러의 컨텍스트 메뉴 중 **[핸들러 업로드]**를 선택하여 핸들러를 DIS에 등록시킨다. 핸들러 업로드 후 WebAdmin의 **[어드민] > [사용자 클래스 설정]** 메뉴의 '**message-validity-class-name**' 항목에서 생성한 핸들러의 이름을 패키지를 포함하여 입력한다.

사용자 클래스 설정

이관관리 핸들러	<input type="text"/>
메세지 유효성 검사 핸들러	<input type="text" value="com.tmax.manual.txGrp_R_tutorial.tx_L_A_tutorial.MsgValidityImpl"/>
전문 공유 핸들러	<input type="text"/>
기관담당자 핸들러	<input type="text"/>

메시지 유효성 검사 핸들러 등록(WebAdmin)

핸들러 등록 후 메시지 에디터의 **[유효성 검사]**를 선택하면 메시지 유효성 검사 결과가 출력된다.



메시지 유효성 검사 결과 - 다이얼로그

메시지 구성 *

+ 추가
- Delete
🔍 검색
🔄 변환
📄 VO Export
🔍 유효성검사

Physical Name	Logical Name	Field Type	Included Str. Name	Included Str. Path	Array Size	Keyword	Comment	Mask	
abc	abc	string							
bcd	bcd	string							
abd	abd	string							
acb	acb	string							

메시지 유효성 검사 결과 - 에디터



메시지 유효성 검사 핸들러를 사용하기 위해서는 스튜디오 컴파일러 버전과 DIS가 설치된

서버의 JAVA 버전이 동일해야 한다. ([Window] > [Preferences] > [Java] > [Compiler] > [Compiler compliance level] : 1.6)

라우팅 핸들러

멀티바인딩에서 바인딩을 위한 핸들러이다. **route** 메소드는 멀티바인딩이 호출될 때 실행되어 리턴되는 값과 같은 값을 가진 서비스가 호출된다.

메시지 콘텐츠를 Value로 하여 라우팅하는 예제이다.

```
package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.MessageContext;
import com.tmax.anylink.api.multibinding.DefaultRoutingHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class routing extends DefaultRoutingHandler {

    private static final Logger logger = Logger.getLogger(routing.class.getName());

    @Override
    public String route(MessageContext arg0) throws AnyLinkException {
        return (String)new String((byte[])arg0.getContent());
    }
}
```

핸들러 작성이 완료되면 멀티 바인딩의 바인딩 설정에서 선택 옵션을 '**HANDLER**'로 설정하고 선택 옵션 추가정보에 핸들러를 검색하여 입력한다.

SSO 로그인 핸들러

SSO 로그인을 위한 핸들러를 작성한다. **login** 메소드는 사용자가 로그인 시 호출되는 메소드로 리턴되는 **SsoHandlerResult**를 이용하여 로그인 성공/실패 여부를 판단한다. **login** 메소드의 첫 번째 인자는 입력한 사용자 아이디이고, 두 번째 인자는 사용자 비밀번호이다.

다음은 로그인하려는 유저의 아이디와 비밀번호를 로그로 출력하고 로그인을 성공시키는 예제이다.

```
package com.tmax.pkg.handlerCase1;

import com.tmax.anylink.api.SsoLogin.DefaultSsoLoginHandler;
import com.tmax.anylink.api.SsoLogin.SsoHandlerResult;
import com.tmax.anylink.api.SsoLogin.SsoHandlerResult.HandlerResultType;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class ssologin extends DefaultSsoLoginHandler {
```

```

private static final Logger logger = Logger.getLogger(ssologin.class.getName());

@Override
public SsoHandlerResult login(String arg0, String arg1)
    throws AnyLinkException {
    logger.info("SSO ID : " + arg0 + ", PW : " + arg1);
    SsoHandlerResult result = new SsoHandlerResult();
    result.setResult(HandlerResultType.SUCCESS);
    return result;
}
}

```

핸들러 작성이 완료되면 배포 후 WebAdmin의 **[관리자] > [DIS 설정]**에서 **SSO 설정**의 '**SSO 방식**' 항목을 'UserClass 방식'으로 설정하고 SSO 핸들러에 작성한 핸들러를 입력한다.

모니터링 이벤트 핸들러

모니터링 이벤트가 발생한 경우 호출되는 핸들러이다. **event** 메소드는 이벤트가 발생하였을때 호출되는 메소드로 첫 번째 인자에 이벤트 타입 정보, 두 번째 인자에 모니터링 이벤트 정보를 맵으로 얻어 올 수 있다. 맵에서 가져올 수 있는 정보는 **[운영관리] > [이벤트 관리] > [이벤트 목록]** 메뉴의 '**이벤트 메시지**' 항목에서 확인할 수 있다.

다음은 이벤트가 발생할 때 발생한 이벤트의 정보를 로그로 출력하는 예제이다.

```

package com.tmax.pkg.handlerCase1;

import java.util.Map;

import com.tmax.anylink.api.event.DefaultMonitoringEventHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;
import com.tmaxsoft.schemas.anylink.MonitoringEventType;

public class MonitoringEvent extends DefaultMonitoringEventHandler {

    private static final Logger logger = Logger.getLogger(MonitoringEvent.class.getName());

    @Override
    public void event(MonitoringEventType arg0, Map<String, Object> arg1)
        throws AnyLinkException {
        logger.info(arg0.toString());
    }
}

```

핸들러 작성이 완료되면 배포 후 WebAdmin의 **[운영관리] > [이벤트 관리] > [이벤트 목록]** 메뉴에서 각 이벤트들의 정보에서 '**이벤트 핸들러**' 항목에 핸들러를 검색 후 설정한 후 저장한다.

전문 관리 핸들러

전문 관리 Admin에서 **[운영관리] > [전문공유]** 메뉴에서 **[전송]** 버튼을 통해 호출되는 핸들러이다. **callHandler** 메소드는 **[전문공유]**의 **[전송]** 버튼을 클릭했을 때 호출되는 메소드로 체크된 메시지 리스트를 인자로 받아 해당 메시지들의 정보를 가져올 수 있다.

다음은 전송할 메시지들의 sysid를 로그로 출력하는 예제이다.

```
package com.tmax.pkg.handlerCase1;

import java.util.List;

import com.tmax.anylink.api.msgtransfer.DefaultMsgTransferHandler;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.dis.msg.ResourceInfoPK;
import com.tmax.anylink.logging.Logger;

public class MsgTransfer extends DefaultMsgTransferHandler {

    private static final Logger logger = Logger.getLogger(MsgTransfer.class.getName());

    public String callHandler(List<ResourceInfoPK> arg0)
        throws AnyLinkException {
        for(int i = 0; i<arg0.size(); i++) {
            ResourceInfoPK resource = arg0.get(i);
            logger.info("resource sysid = " + resource.getSysId());
        }
        return null;
    }
}
```

핸들러 작성이 완료되면 배포 후 WebAdmin의 **[관리자] > [DIS 설정]** 메뉴에서 '**전문 공유 핸들러**' 항목에 작성한 핸들러 클래스를 입력한다. WebAdmin의 **[운영관리] > [전문 공유]** 메뉴에서 전송할 메시지들을 체크한 뒤 **[전송]** 버튼을 클릭하여 핸들러를 호출할 수 있다.

기관 담당자 핸들러

WebAdmin의 **[운영관리] > [대외 연락처] > [담당자]** 메뉴에서 **[전송]** 버튼을 통해 호출되는 핸들러이다. **callHandler** 메소드는 **[담당자]** 메뉴에서 **[전송]** 버튼을 클릭했을 때 호출되는 메소드로 체크된 담당자들의 정보를 가져올 수 있다.

다음은 선택한 담당자들의 이름을 로그로 출력하는 예제이다.

```
package com.tmax.pkg.handlerCase1;

import java.util.List;

import com.tmax.anylink.api.partneraddress.DefaultPartnerAddressHandler;
import com.tmax.anylink.api.partneraddress.PartnerUser;
import com.tmax.anylink.common.AnyLinkException;
import com.tmax.anylink.logging.Logger;

public class ParnerAddress extends DefaultPartnerAddressHandler {
```

```

private static final Logger logger = Logger.getLogger(PartnerAddress.class.getName());
@Override
public String callHandler(List<PartnerUser> partnertUserList)
    throws AnyLinkException {
    for(int i = 0; i<partnertUserList.size(); i++) {
        PartnerUser user = partnertUserList.get(i);
        logger.info("User Name = " + user.getUserName());
    }
    return super.callHandler(partnertUserList);
}
}

```

핸들러 작성이 완료되면 배포 후 WebAdmin의 **[관리자] > [DIS 설정]** 메뉴에서 '**기관담당자 핸들러**' 항목에 작성한 핸들러 클래스를 입력한다. WebAdmin의 **[운영관리] > [대외 연락처] > [담당자]** 메뉴에서 호출할 담당자들을 체크한 뒤 **[전송]** 버튼을 클릭하여 핸들러를 호출할 수 있다.

코릴레이션 핸들러

코릴레이션 값 설정을 위한 핸들러이다. **matchCorrelation** 메소드는 응답 메시지에 대하여 코릴레이션 값을 설정하여 반환하고 **setCorrelationValue** 메소드는 요청 메시지에 대하여 코릴레이션 값을 설정하여 반환한다.

다음은 코릴레이션 값 앞에 "1234"를 추가하는 핸들러 예제이다.

```

package com.txTEST;

import com.tmax.anylink.api.serviceflow.DefaultCorrelationHandler;
import com.tmax.anylink.logging.Logger;

public class CorrHandlerEx extends DefaultCorrelationHandler {

    private static final Logger logger = Logger.getLogger(CorrHandlerEx.class
        .getName());

    @Override
    public String matchCorrelation(String recvMsg) {
        // TODO Auto-generated method stub
        String data = "1234";
        String resCorrVal = data + recvMsg;
        logger.finest("##[Response]Correlation value : " + resCorrVal);

        return resCorrVal;
    }

    @Override
    public String setCorrelationValue(String sendMsg) {
        // TODO Auto-generated method stub
        String data = "1234";
        String reqCorrVal = data + sendMsg;
        logger.finest("##[Request]Correlation value : " + reqCorrVal);
        return reqCorrVal;
    }
}

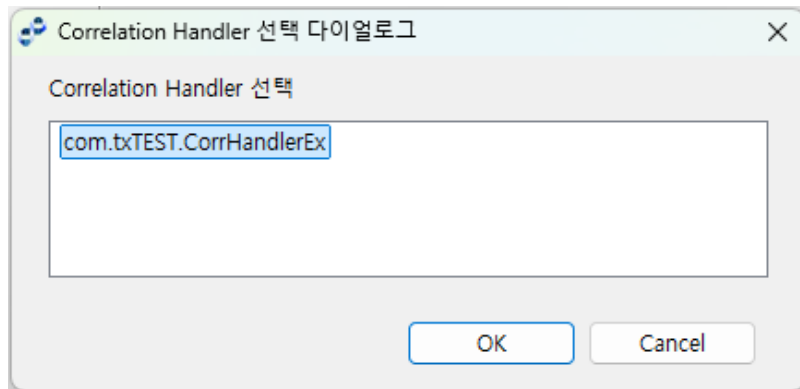
```


핸들러 작성이 완료되면 **[메시지 이벤트] > [Event Preference]** 화면의 **[기본정보]** 탭의 **'Correlation Handler'** 항목에 핸들러를 입력하여 사용한다. 핸들러 입력창에 직접 입력 시 패키지를 포함한 핸들러의 이름을 입력한다.

Correlation Handler

코릴레이션 핸들러 설정

[Search] 버튼 클릭 시 **[Correlation Handler 선택 다이얼로그]**를 통해 사용할 코릴레이션 핸들러를 선택할 수 있다.



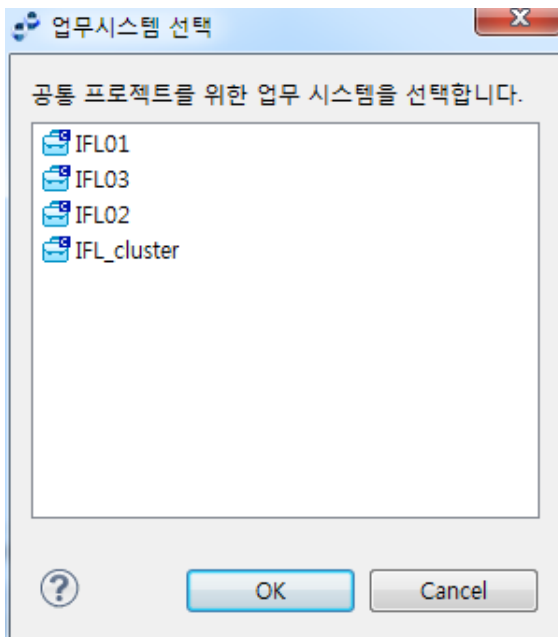
코릴레이션 핸들러 설정 - 다이얼로그

8. 공통업무

본 장에서는 AnyLink 스튜디오의 공통업무 기능과 사용법에 대해 설명한다. 공통업무는 동일한 업무 시스템을 가지는 프로젝트에 대해서 공통으로 사용할 수 있는 리소스들의 집합이다.

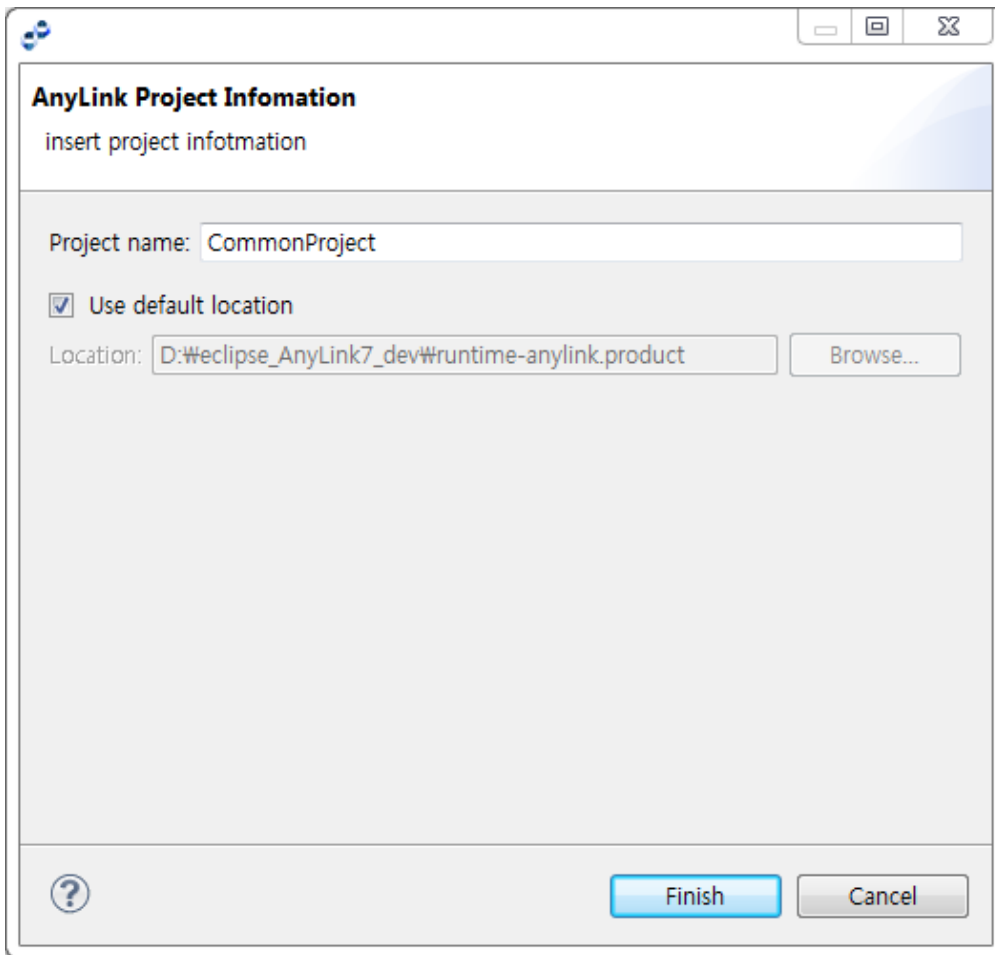
8.1. 공통업무 프로젝트 생성

공통업무 프로젝트의 생성은 왼쪽 거래 네비게이터의 컨텍스트 메뉴에서 **[새로만들기] > [공통업무 프로젝트]**를 선택한다. **업무시스템 선택 화면**에서 공통 프로젝트를 위한 업무 시스템을 선택하고 **[OK]** 버튼을 클릭한다. 업무시스템당 한 개의 공통업무 프로젝트만 생성 가능하다.



업무시스템 선택 화면

프로젝트 생성 화면에 각 항목을 입력하고 **[Finish]** 버튼을 클릭한다.



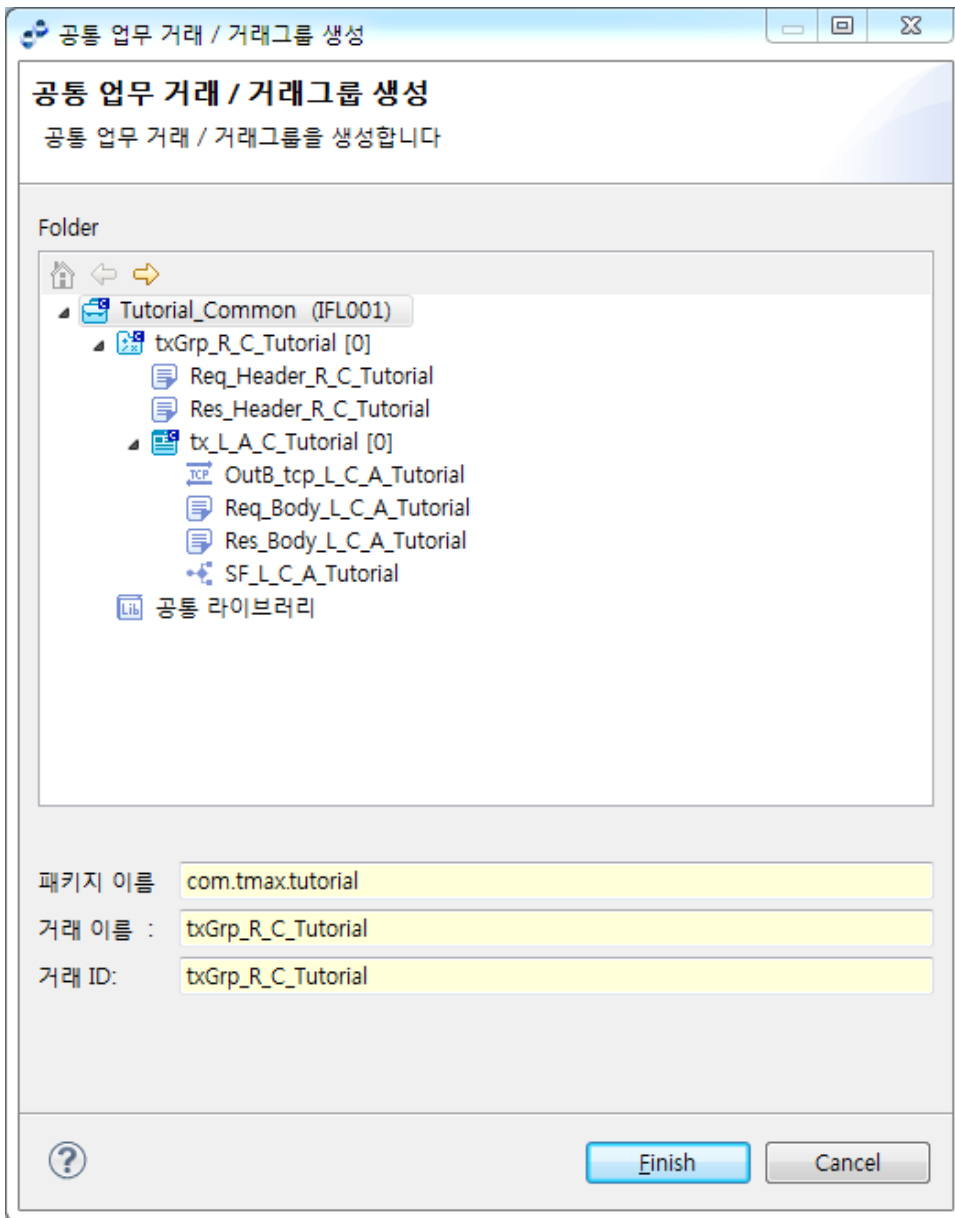
프로젝트 생성 화면

항목	설명
Project name	프로젝트 이름을 입력한다. 단, 폴더 이름으로 특수문자(?, <, >, ", *,)는 쓸 수 없다.

8.2. 공통업무 거래(그룹) 생성

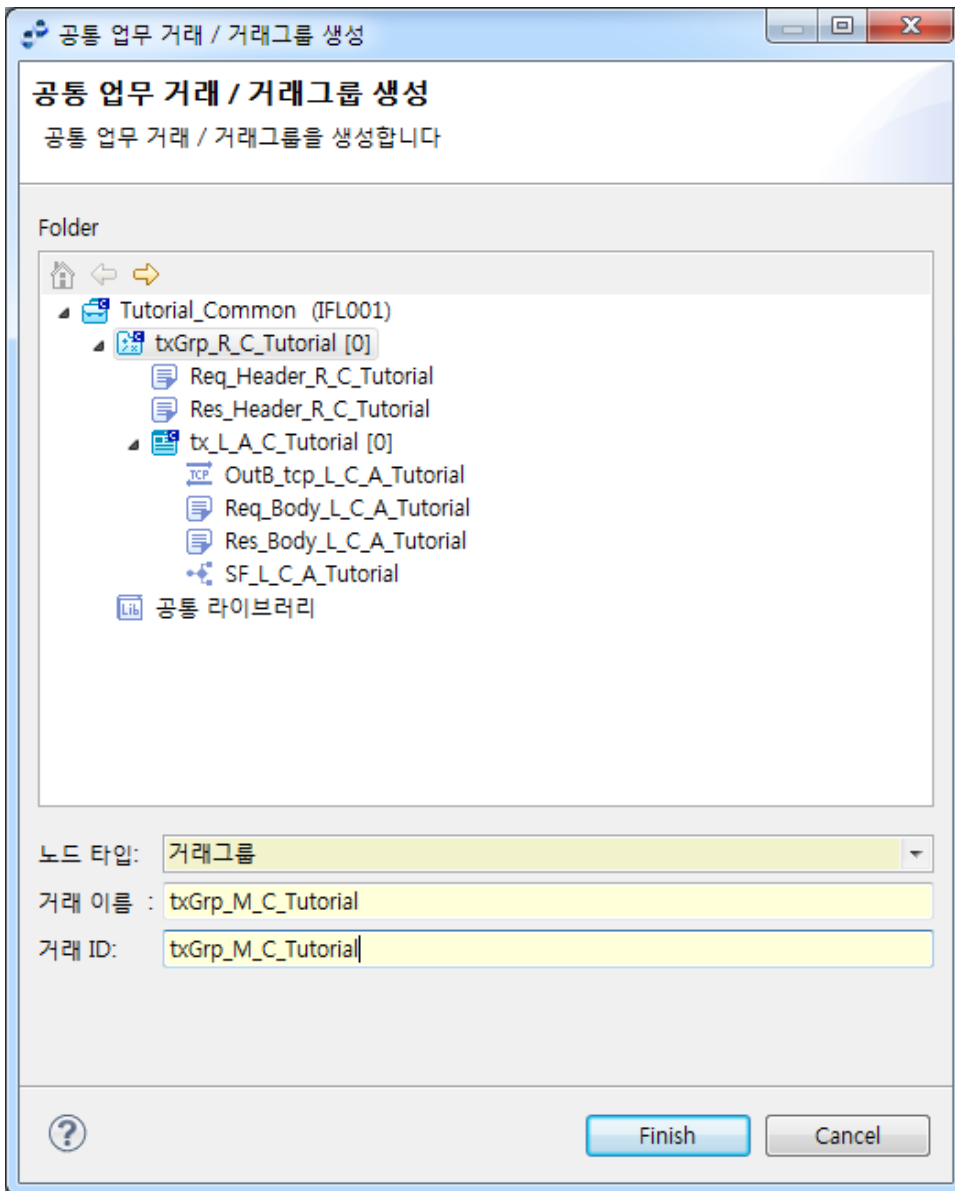
공통업무 거래/거래그룹은 일반 업무 거래/거래그룹과 동일하게 노드 타입에 따라 생성 방법이 구분된다.

최상위 거래그룹의 경우 네비게이터에서 **공통업무 프로젝트**의 컨텍스트 메뉴에서 **[새로만들기] > [공통업무 거래/거래그룹]**을 선택한다.



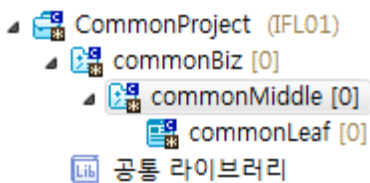
공통업무 거래/거래그룹 생성 화면 - 최상위 거래그룹(Root)

거래/거래그룹의 경우 **공통업무 거래그룹** 리소스의 컨텍스트 메뉴에서 **[새로만들기] > [공통업무 거래/거래그룹]**을 선택해서 생성이 가능하다. 생성 화면의 각 항목은 일반 거래/거래그룹의 항목과 동일하다.



공통업무 거래/거래그룹 생성 화면 - 거래그룹(Middle), 거래(Leaf)

공통업무 거래그룹 및 거래가 생성되면 네비게이터에서 생성된 거래를 확인할 수 있다.



공통업무 생성 확인

에디터 화면은 거래/거래그룹과 동일하므로 **거래/거래그룹 에디터**를 참고한다. 공통업무 프로젝트에서는 멀티바인딩, WSDL 생성은 할 수 없다.

9. 배포 및 배포해제

본 장에서는 DIS에 리소스를 등록하는 방법과 리소스를 등록해제하는 방법을 설명한다.

9.1. 개요

AnyLink는 단순히 거래/거래그룹에 속한 리소스를 등록하고 다운로드하는 기능만 제공하는 것이 아니라 버저닝을 통해 필요한 경우 리소스들을 롤백(Rollback)하는 기능도 제공한다.

AnyLink에서 DIS에 거래/거래그룹에 리소스를 등록하고 런타임 서버에도 리소스를 등록하는 것을 배포한다고 정의하며 동일한 거래/거래그룹의 리소스를 수정하여 다시 올려 배포하는 것은 재배포, DIS, 런타임 서버에서에서 배포된 리소스를 제거하는 작업은 **배포해제**라고 한다.

배포된 리소스는 런타임 서버에도 등록되어 사용이 가능해지는데, 대부분의 리소스의 경우 런타임 서버가 대부분 인지하고 있으므로 특별한 배포는 런타임 서버 엔진 수행에 영향이 없으나 공통 라이브러리 항목은 배포가 되기 전까지 런타임 서버 엔진에서 수행되지 않는다. 각 내용에 대한 자세한 설명은 해당하는 항목을 참조하도록 한다.

9.2. 리소스 배포(재배포)

리소스를 배포하는 경우 배포하는 리소스는 IAR이라고 하는 zip 아카이브 형태의 압축 파일로 압축되어 등록된다. 최초로 등록된 경우 리소스는 버전 1로 기록되어 압축 파일이 저장되며, 등록되지 않은 경우는 버전 0으로 간주된다.

재배포를 할 경우 리소스의 버전은 하나씩 증가하며 사용자는 원하는 버전으로 각 리소스를 롤백시킬 수 있다. 이 때 주의해야 할 사항으로는 리소스를 롤백한 후 수정하여 서버에 배포를 요청할 경우 서버의 최신 버전과 로컬의 버전이 달라 배포 요청을 DIS가 받아들이지 않는다는 점이다. 이 경우 **'버전 점검 안함'** 항목을 활성화하여 현재 로컬의 버전을 최신 버전으로 등록할 수 있으나 기존의 최신 버전에 커다란 문제가 생기지 않는 한 이런 방식의 사용은 권장하지 않는다.

9.2.1. 배포 단위

배포는 거래/거래그룹 단위로 리소스를 묶어 처리하며, 거래그룹과 거래를 기준으로 내부에 속한 모든 리소스들을 배포할 수 있다. 버저닝 또한 거래/거래그룹 단위로 처리되며 거래그룹 단위인 경우 하위 거래그룹과 하위 거래를 같이 배포하지 않는 것이 기본으로 되어 있다. 만일 하위 거래들까지 같이 배포하고 싶을 경우에는 해당하는 옵션을 설정한다.

9.2.2. 배포 과정

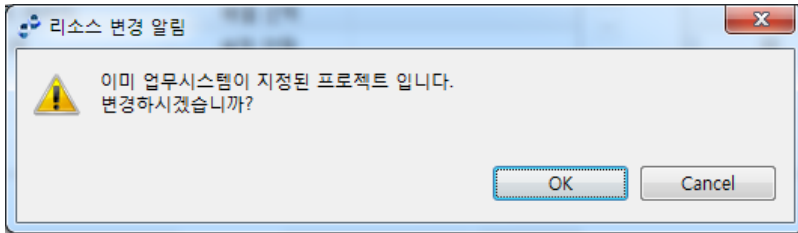
본 절에서는 배포에 필요한 업무시스템 할당과 실제 배포를 수행하는 과정을 나누어 설명한다.

업무시스템 할당


리소스를 배포하기 위해서는 먼저 프로젝트를 업무시스템에 할당하는 과정이 필요하므로 프로젝트를 업무시스템에

할당하는 방법에 대해 설명한다.

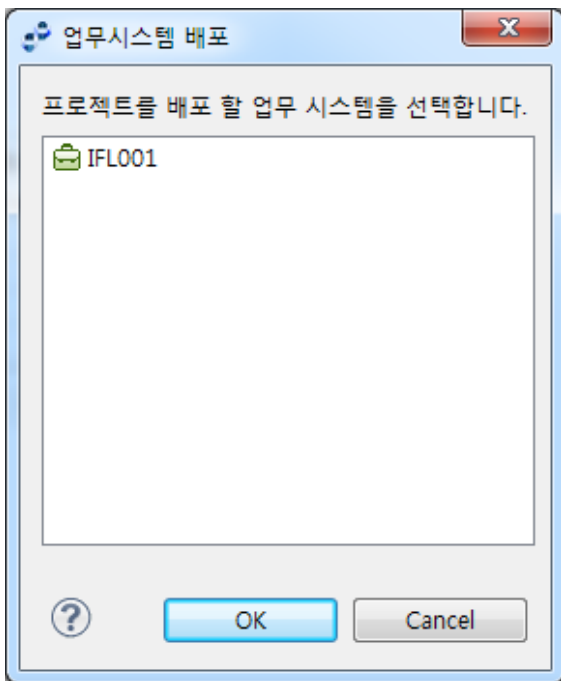
왼쪽 네비게이터 프로젝트의 컨텍스트 메뉴에서 **[업무시스템 할당]**을 선택해서 프로젝트에 업무시스템을 할당한다. 만일 업무시스템이 지정되어 있는 경우 다음과 같은 메시지가 출력된다. **[OK]** 버튼을 클릭하면 업무시스템 할당으로 넘어간다.



업무시스템 할당할 경우 경고 메시지

프로젝트가 업무시스템에 할당되었는지의 여부는 왼쪽의 **프로젝트 네비게이터**를 보면 알 수 있다. 프로젝트가 업무시스템에 할당된 경우 다음과 같이 프로젝트 이름 옆에 업무시스템 이름이 "프로젝트 이름 (업무시스템)"  Tutorial (IFL001) 으로 표기된다.

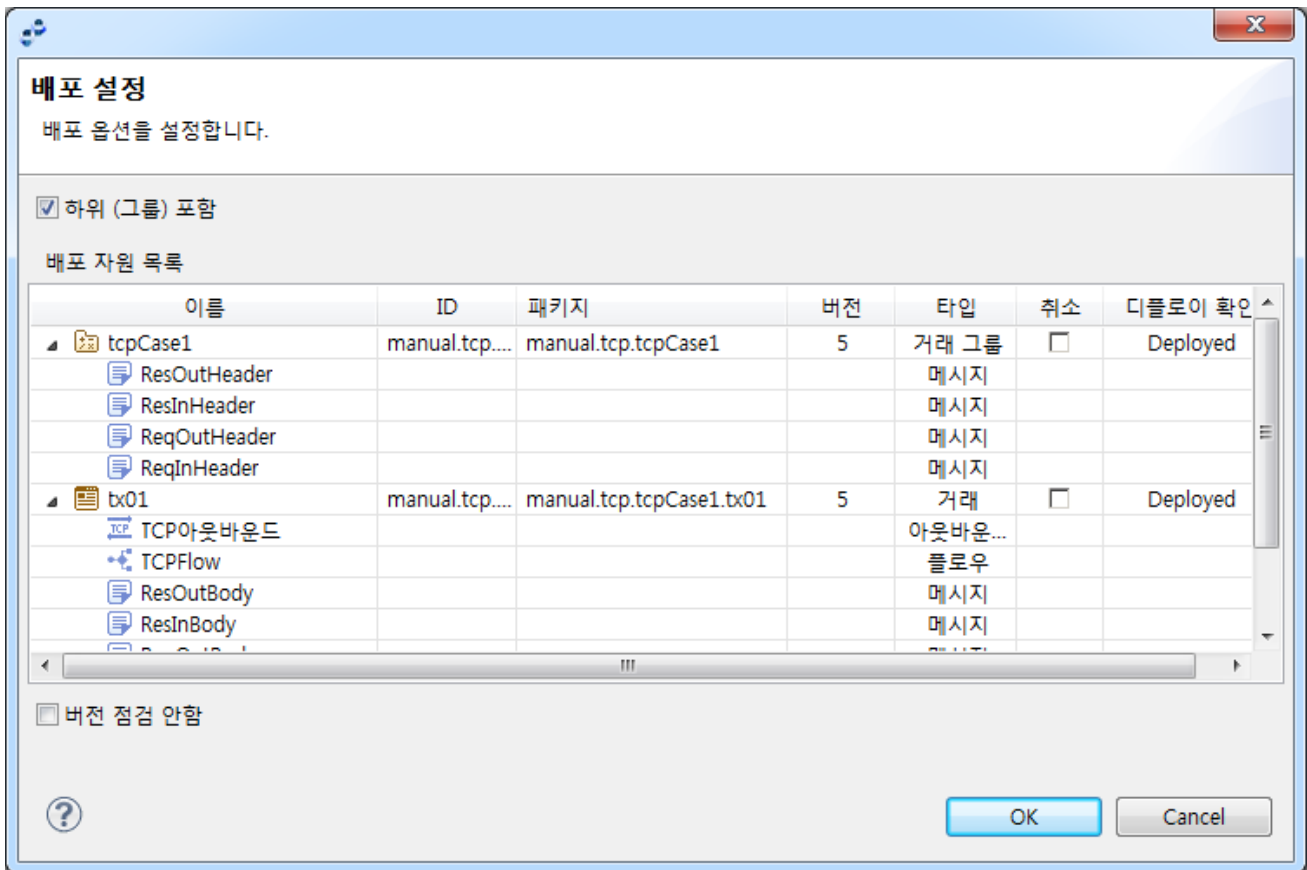
업무시스템 배포 화면에서 배포할 업무시스템을 선택한 후 **[OK]** 버튼을 클릭해서 프로젝트를 업무시스템에 할당한다.



업무시스템 배포 다이얼로그

리소스 배포

리소스를 배포하려면 배포하려는 거래들을 선택하고, 네비게이터 거래/거래그룹 컨텍스트 메뉴에서 **[배포]**를 선택한다. **배포 설정 화면**에서 각 항목을 설정한 후 **[OK]** 버튼을 클릭하면 리소스가 DIS에 전송되고 런타임 서버에서 리소스가 등록되어 배포가 완료된다. **Ctrl** 키를 누른채 여러 거래를 선택하여 **[배포]**하면 여러 거래를 배포할 수 있다.



배포 설정 화면

항목	설명
하위 (그룹) 포함	해당 거래그룹에 속한 하위 거래그룹과 거래를 함께 묶어 배포할 것인지를 선택한다. 선택할 때 모든 하위 거래그룹과 거래가 한 번에 배포된다.
배포 자원 목록	<p>배포할 리소스의 목록을 조회한다. 배포 자원 목록에서 배포할 자원이 맞는지 점검하고 배포한다.</p> <ul style="list-style-type: none"> ◦ 이름 : 리소스 이름이다. ◦ ID : 거래 노드의 SYS ID이다. ◦ 패키지 : 거래 노드의 패키지명이다. ◦ 버전 : 거래 노드의 현재 버전이다. ◦ 타입 : 리소스 타입이다. ◦ 취소 : 체크박스를 선택할 때 해당 노드의 리소스는 배포에서 제외된다. ◦ 디플로이 확인 : 배포된 이력이 있는 거래의 경우 DIS에서의 배포 상태를 보여준다.
버전 점검 안 함	현재 리소스를 최신 버전으로 만드는 경우에 사용한다.

9.3. 리소스 배포해제

리소스를 배포해제하는 경우 DIS에서 리소스와 관련된 모든 데이터를 배포해제 상태로 변경한다. 본 절에서는 리소스 배포를 해제하는 방법에 대해 설명한다.

항목	설명
배포해제 자원 목록	<p>배포해제할 리소스의 목록을 조회한다. 배포 자원 목록에서 배포할 자원이 맞는지 점검하고 배포한다.</p> <ul style="list-style-type: none"> ◦ 이름 : 리소스 이름이다. ◦ ID : 거래 노드의 SYS ID이다. ◦ 패키지 : 거래 노드의 패키지명이다. ◦ 버전 : 거래 노드의 현재 버전이다. ◦ 타입 : 리소스 타입이다. ◦ 디플로이 확인 : 배포된 이력이 있는 거래의 경우 DIS에서의 배포 상태이다.
버전 점검 안 함	현재 로컬 리소스가 최신이 아니지만 배포해제를 요구하는 경우에 사용한다.

2. 업무시스템 할당해제

왼쪽 네비게이터 프로젝트의 컨텍스트 메뉴에서 **[업무시스템 할당해제]**를 선택해서 프로젝트에 업무시스템을 할당해제한다. 공통업무 프로젝트의 경우 할당을 해제할 수 없으며 오직 업무시스템의 할당을 변경하는 동작만 가능하다. 해당 동작은 [배포 과정](#)을 참고한다.

9.4. 공통 라이브러리 배포 및 배포해제

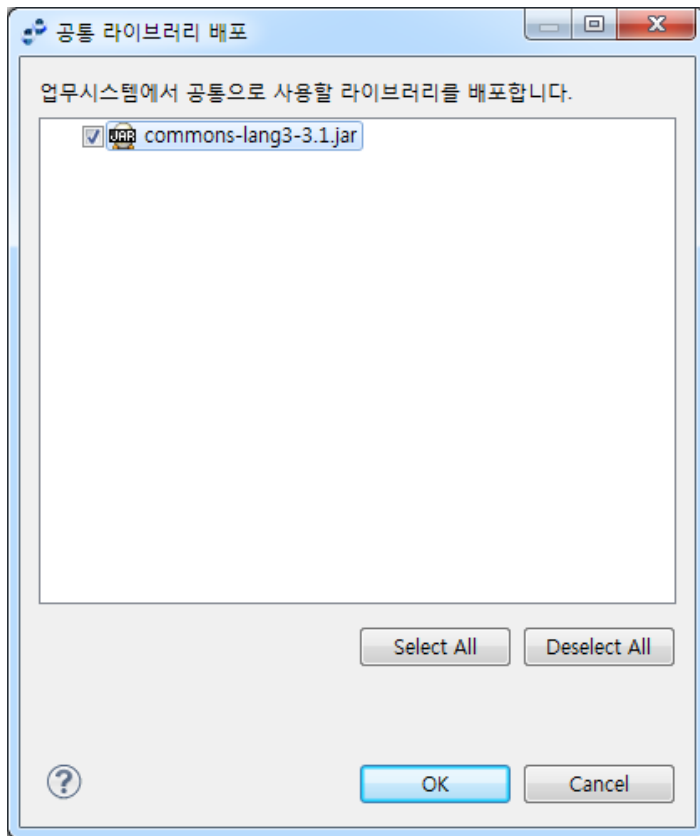
본 절에서는 라이브러리를 배포하는 방법에 대해 설명한다.

공통 라이브러리는 Java로 컴파일된 라이브러리 파일들로 jar 파일을 의미한다. jar 파일을 리소스로 배포하여 런타임 서버에 등록되면 모든 거래에서 해당 라이브러리를 사용할 수 있도록 하므로, 다양한 클래스의 기능을 유연하게 사용할 수 있도록 만들어 준다. 단, 런타임 서버 내의 라이브러리와 충돌하는 경우 런타임 서버가 어떤 라이브러리를 호출해야 하는지 제대로 구분하지 못해 AnyLink의 수행에 문제가 생길 수 있으니 주의하여 설정하도록 한다.

공통 라이브러리를 사용하기 위해서는 공통 라이브러리가 배포되어 저장될 경로를 미리 설정해야 한다. 경로는 WebAdmin의 **[어드민] > [DIS 설정]** 메뉴의 '**공통 라이브러리 경로**' 항목에서 설정 가능하다.

9.4.1. 공통 라이브러리 배포

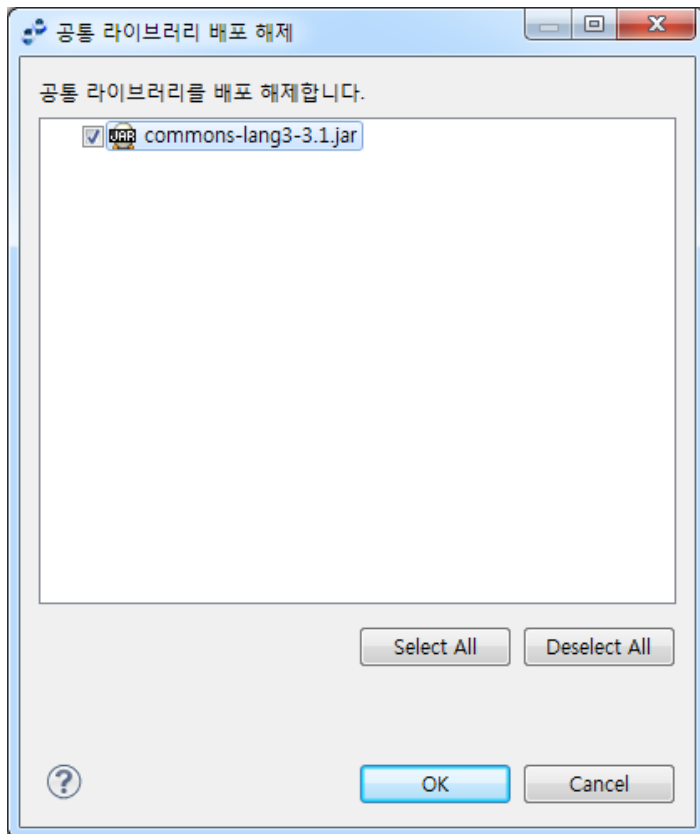
공통 라이브러리를 배포하려면 왼쪽 네비게이터에서 **공통 라이브러리**의 컨텍스트 메뉴에서 **[배포]**를 선택한다. **공통 라이브러리 배포 화면**에서 배포할 라이브러리의 체크박스를 선택한 후 **[OK]** 버튼을 클릭한다.



공통 라이브러리 배포 다이얼로그

9.4.2. 공통 라이브러리 배포해제

공통 라이브러리를 배포해제하려면 왼쪽 프로젝트 네비게이터에서 **공통 라이브러리**를 선택한 후 컨텍스트 메뉴에서 **[배포해제]**를 선택한다. **공통 라이브러리 배포 화면**에서 배포해제할 라이브러리의 체크한 후 **[OK]** 버튼을 클릭한다.



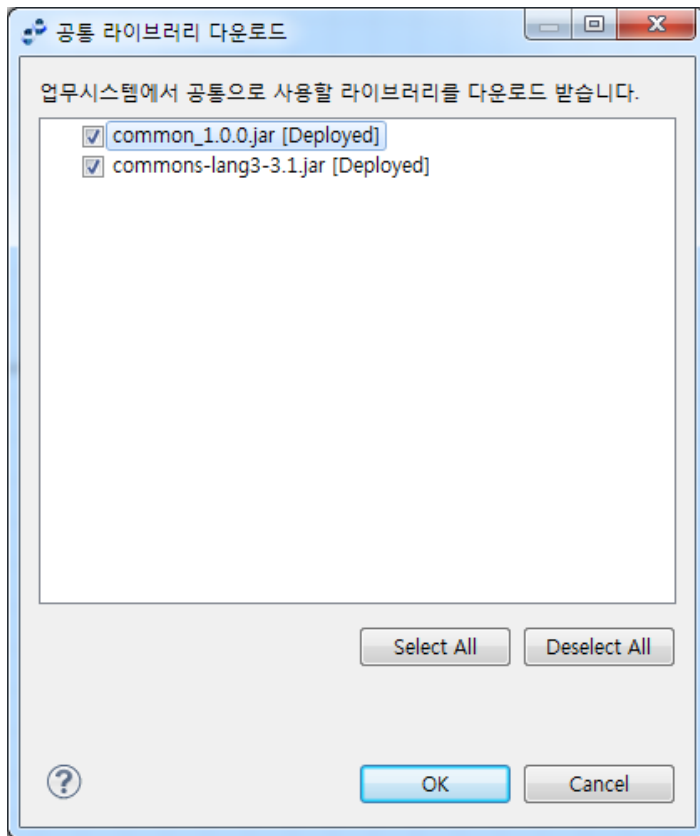
공통 라이브러리 배포해제 다이얼로그

9.4.3. 라이브러리 추가

왼쪽 프로젝트 네비게이터에서 **공통 라이브러리**를 선택한 후 컨텍스트 메뉴에서 **[라이브러리 추가]**를 선택하면 공통 라이브러리를 선택할 수 있는 화면이 나타난다. 확장자를 jar로 갖는 파일을 선택 후 **[열기]** 버튼을 클릭하면 공통 라이브러리에 라이브러리가 추가된다.

9.4.4. 라이브러리 다운로드

왼쪽 프로젝트 네비게이터에서 **공통 라이브러리**를 선택한 후 컨텍스트 메뉴에서 **[라이브러리 다운로드]**를 선택하면 DIS에 배포되어 있는 공통 라이브러리 리스트가 나타난다. 다운로드할 라이브러리를 선택한 후 **[OK]** 버튼을 클릭한다.



라이브러리 다운로드

10. 리소스 및 다운로드

본 절에서는 시스템에서 사용하는 각종 리소스들을 검색하고 다운로드하는 방법을 설명한다.

10.1. 개요

AnyLink는 거래, 거래그룹, 유저 클래스, 어댑터 등 실제 엔진에서 수행되거나 엔진에 정보를 주는 모든 요소를 리소스로 정의하며 AnyLink의 사용자는 WebAdmin과 Studio에서 각종 리소스를 생성하고 등록할 수 있다. AnyLink는 이러한 리소스들의 효율적인 관리를 위하여 거래/거래그룹으로 리소스들을 묶고 DIS를 통해 버전으로 관리하는 기능을 제공한다.

배포된 리소스는 런타임 서버에도 등록되어 사용이 가능한데 대부분의 리소스의 경우 런타임 서버가 대부분 인지하고 있으므로 특별한 배포는 런타임 서버 엔진 수행에 영향이 없으나 공통 라이브러리 항목은 배포되기 전까지 런타임 서버 엔진에서 수행되지 않는다. 이 외에도 배포된 리소스는 사용자들이 이미 정의된 리소스들을 DIS 서버에 등록하여 여러 관리자가 동시에 같은 리소스를 다운로드 받아 사용할 수 있도록 지원한다.

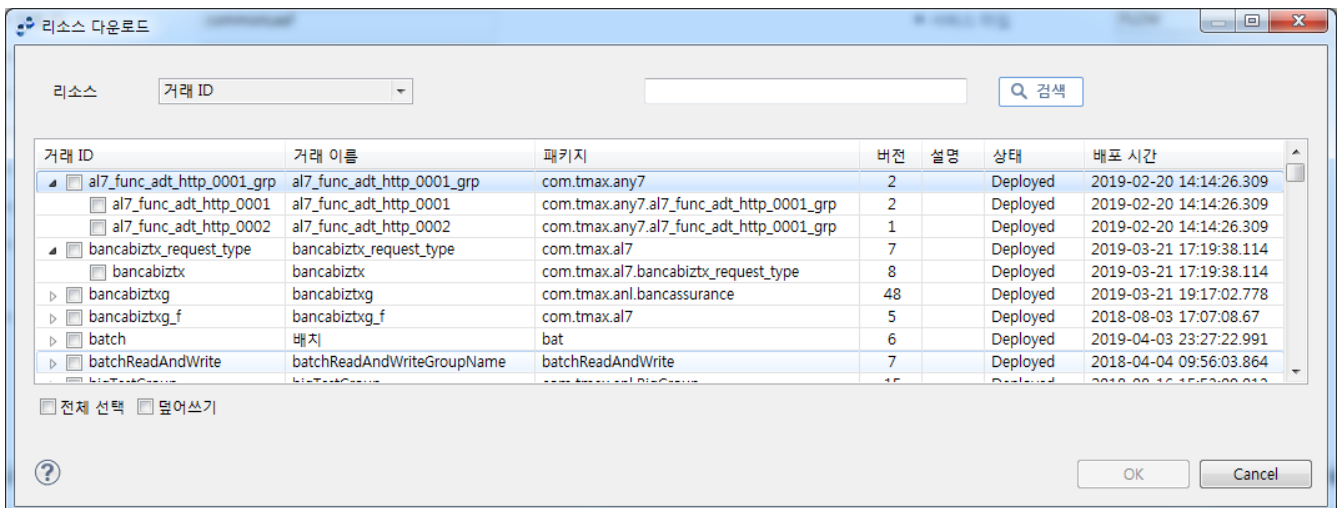
본 절에서는 이와 같은 기능을 사용하는 방법을 설명하도록 하며, 실제 리소스를 DIS에 등록하고 버저닝하는 기능은 [배포 및 배포해제](#)를 참고한다.



리소스에 대한 자세한 설명은 "AnyLink 런타임 엔진 서버 안내서"를 참고한다.

10.2. 리소스 검색 및 다운로드

리소스의 검색과 다운로드는 왼쪽 네비게이터에서 프로젝트, 거래그룹, 거래 중에 하나를 선택한 후 컨텍스트 메뉴에서 **[다운로드]**를 선택한다. **리소스 다운로드 화면**에서 리소스를 찾아 체크를 한 후 **[OK]** 버튼을 클릭한다.



리소스 다운로드 다이얼로그

항목	설명
리소스	검색할 리소스의 기준을 설정한다. 거래 이름, 거래 아이디, 패키지를 기준으로 검색을 지원한다. 항목 뒤의 텍스트박스에 검색할 키워드를 입력하고 [검색] 버튼을 클릭하여 검색을 수행할 수 있다.
리소스 리스트	<p>조건에 맞는 리소스의 목록이 나타난다. 만일 계층구조가 있는 리소스의 경우 이름 옆에 [▷] 버튼이 생성되며, 클릭할 경우 하위 리소스들을 조회할 수 있다.</p> <p>다음은 리스트의 컨텍스트 메뉴에 대한 설명이다.</p> <ul style="list-style-type: none"> ◦ [상위거래 포함] : 선택한 리소스 상위 노드의 거래그룹을 포함하여 다운로드할 경우 사용한다. 상위 노드의 거래그룹이 체크된다. ◦ [하위거래 포함] : 선택한 리소스 하위 노드의 거래/거래그룹을 포함하여 다운로드할 경우 사용한다. 하위 노드의 거래/거래그룹이 체크된다. ◦ [선택그룹 해제] : 선택한 리소스가 포함된 상/하위 노드 거래/거래그룹의 체크박스가 모두 해제된다. <p>버전 컬럼을 선택할 때 [▽] 버튼이 생성되며 버튼을 클릭할 때 하위 버전들의 리스트가 나타난다. 리스트에서 버전을 선택하면 해당 버전이 다운로드되어 버전 롤백(Rollback)이 가능하다.</p>
전체 선택	리소스 리스트에 나타난 모든 항목을 다운로드하기 위해 선택하는 경우 사용한다.
덮어쓰기	동일한 리소스가 있는 경우 로컬에 있는 리소스를 DIS의 리소스로 덮어쓰고 싶을 경우 체크한다.

부록 A: 다이얼로그

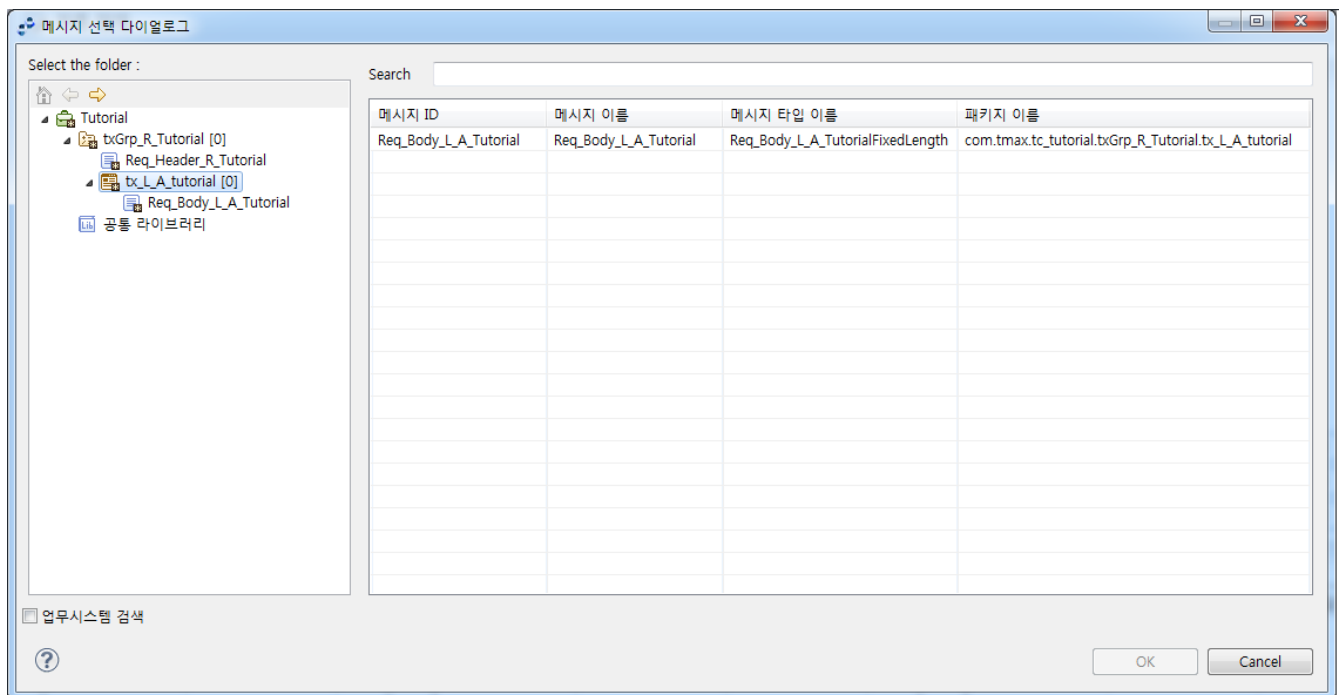
본 부록에서는 스튜디오에서 생성되는 다이얼로그에 대해 설명한다. 스튜디오를 사용하면서 특정 다이얼로그를 참조하는 경우 이 장의 다이얼로그 목록에서 설명을 찾아 참고한다. 본 부록에서 설명하는 다이얼로그는 여러 장에서 참조하는 일반적인 다이얼로그로, 특정 장에서만 나타나는 다이얼로그는 해당 장에서 설명한다.

A.1. 메시지 관련 다이얼로그

A.1.1. 메시지 선택 다이얼로그

메시지 필드 선택 다이얼로그는 원하는 메시지를 선택해야 할 경우 나타난다. 왼쪽 **프로젝트 네비게이터**에서 원하는 **거래/거래목록**을 선택하면 오른쪽 **메시지 리스트**에 거래/거래목록에 속한 모든 메시지의 목록이 나타난다. 목록에서 원하는 메시지를 찾아 클릭한 후 아래의 **[OK]** 버튼을 클릭하면 해당 메시지가 선택된다.

'Search' 항목을 사용해서 메시지를 이름으로 검색할 수도 있다.

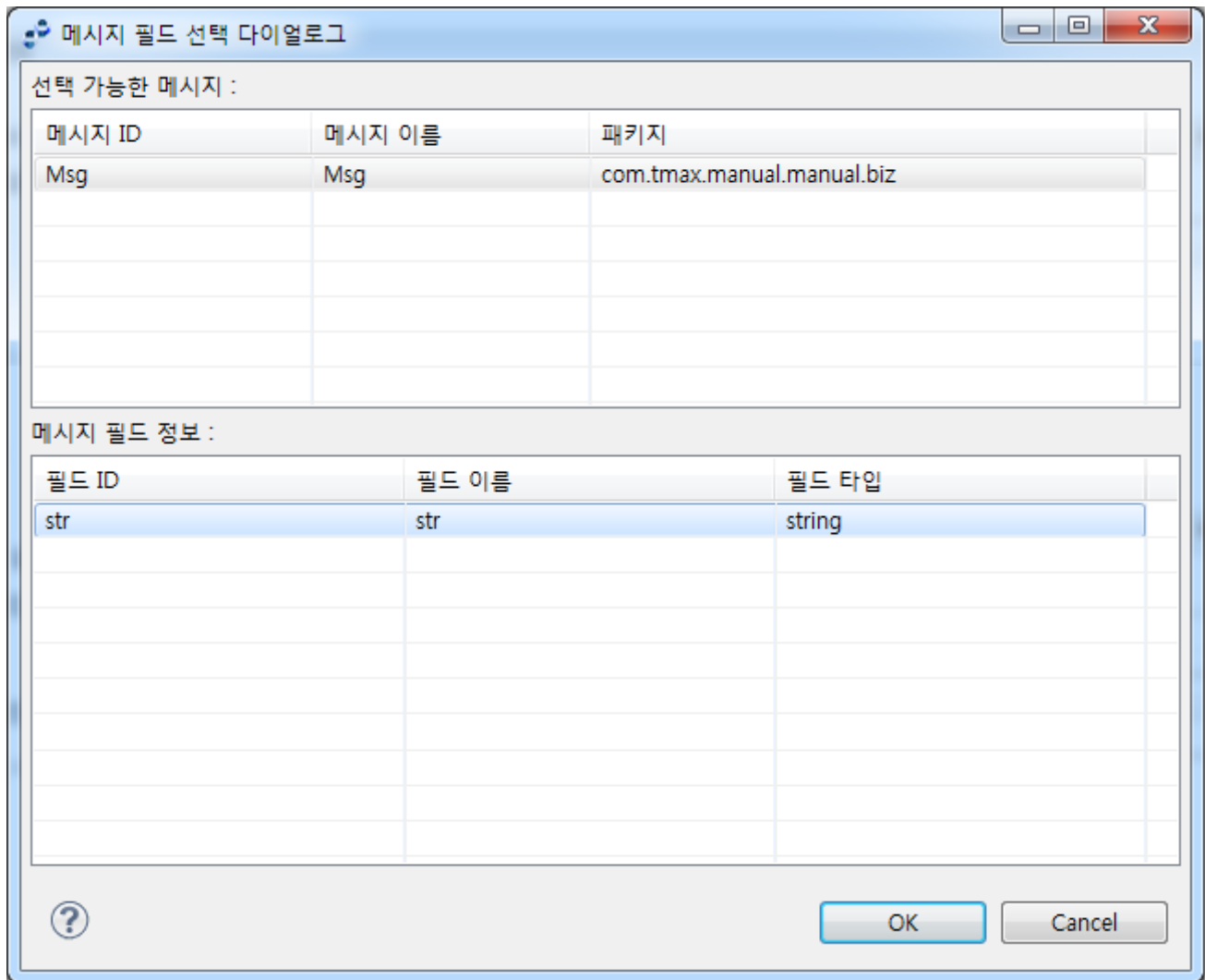


메시지 선택 다이얼로그

A.1.2. 메시지 필드 설정 다이얼로그

메시지 필드 선택 다이얼로그는 선택한 메시지에 속한 필드 중 하나를 선택해야 하는 경우 나타난다.

선택 가능한 메시지는 현재 작업하는 프로젝트에 속한 모든 메시지들을 보여주므로 원하는 메시지를 골라 클릭하여 선택한다. 메시지를 선택하고 나면 **메시지 필드 정보**에 필드 값이 리스트로 나타나므로 원하는 필드를 선택한 후 **[OK]** 버튼을 클릭하여 선택한 필드 정보를 입력한다.



메시지 필드 선택 Wizard

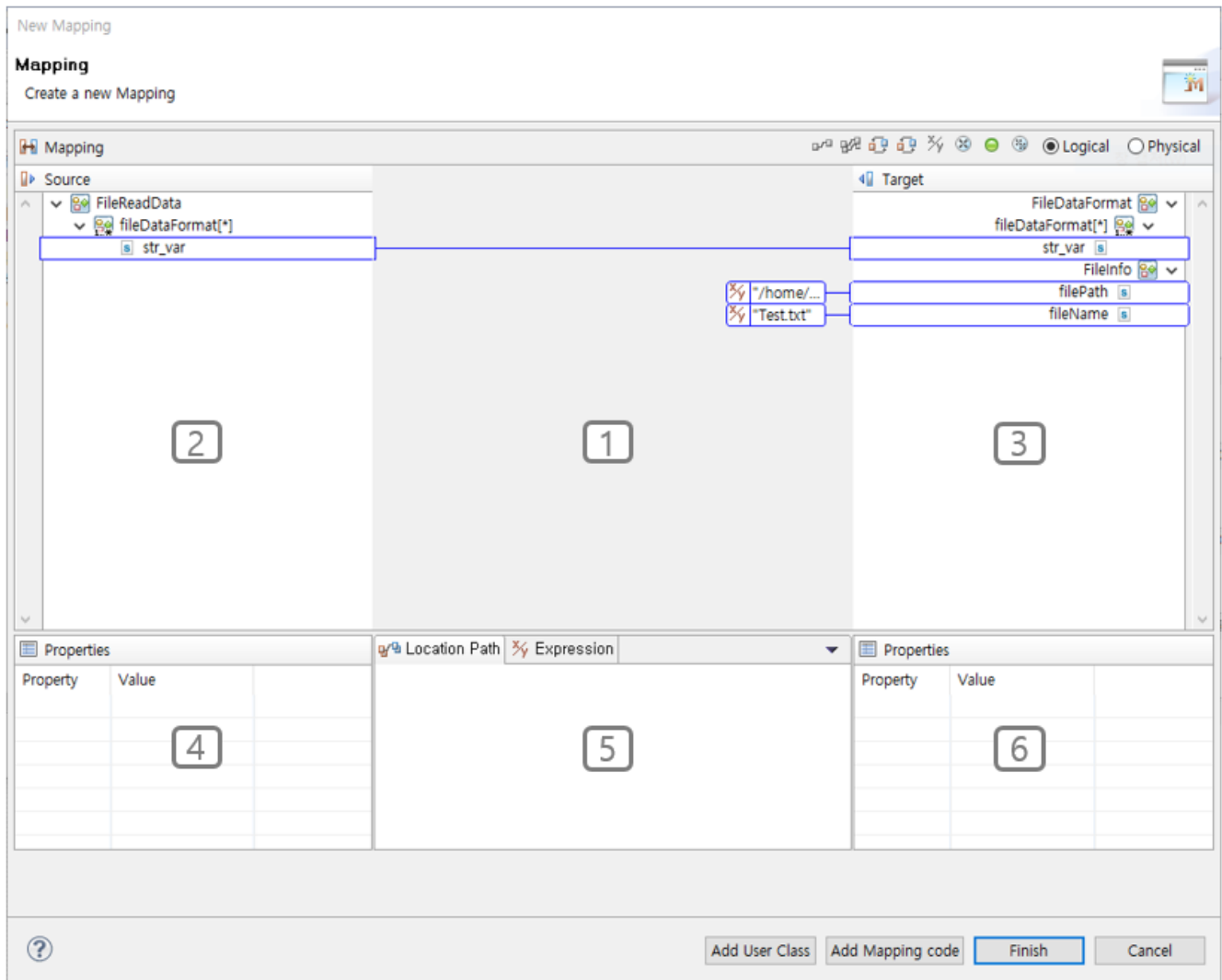
A.2. 매핑 관련 다이얼로그

매핑은 전문을 변환하기 위해 소스에서 타겟의 관계를 정의하는것을 말한다. 매핑은 AnyLink에서 빈번히 일어나는 작업이며, 대부분의 경우 매핑을 정의하기 위해 매핑 마법사를 사용해야 한다.

AnyLink는 다양한 리소스를 통합하기 위해 여러 가지 어댑터를 제공한다. 각종 어댑터는 어댑터가 작동하는 원래의 리소스 구조를 AnyLink 서버가 사용하는 메시지로 변환하며, 이때 변환 과정을 정의하는데 **매핑 마법사**를 사용한다. **매핑 마법사**는 변환하는 리소스의 종류에 상관없이 동일하며, 리소스의 종류에 영향을 받는 부분은 제한되어 있고, 어댑터나 플로우 등 매핑이 상용되는 곳은 모두 사용된다.

A.2.1. 매핑 마법사

다음은 매핑 마법사 **New Mapping 화면**에 대한 설명이다. 매핑 마법사는 크게 매핑 관련 영역(1, 2, 3)과 정보조회/편집 영역(4, 5, 6)으로 나뉜다.






매핑 화면

매핑 관련 영역

소스 트리(2)와 매핑 라인(1) 그리고 타겟 소스(3)로 구성된다.

다음은 매핑 관련 영역 상단 bar의 아이콘과 하위 버튼에 대한 설명이다.

항목	설명
 (Creating mapping)	새로운 매핑을 생성한다.
 (Creating mappings)	
 (Generate mapping automatically /Generate mappings with physical name automatically)	<p>이름이 같은 노드 간에 자동으로 매핑할 수 있다. 소스나 타겟에 이름이 같은 노드가 여러 개 있을 경우 먼저 나오는 노드의 이름이 매핑되므로 자동으로 매핑한 후에는 올바르게 매핑이 되었는지 확인할 필요가 있다.</p> <p>[Generate mappings with physical name automatically] 메뉴를 이용하여 Structure의 경우 매핑할 때 보이는 값은 논리명이므로 물리명을 이용해서 자동 매핑을 한다.</p>

 (Create new Expression)	<p>함수 매핑이나 사칙연산을 정의하는 메뉴이다.</p> <p>타겟에 대하여 이미 정의된 매핑이 있을 경우 [Create new expression] 메뉴는 비활성화되며 함수 매핑 생성은 불가능하다. 사용자는 매핑 마법사의 톨바에서 [Create new expression] 메뉴를 선택하여 트리에 선택된 요소에 대해 함수 매핑을 생성 할 수있다.</p> <p>함수 매핑의 대상이 되는 타겟의 요소는 둥근 사각형으로 둘러싸여 보여진다. 함수는 매핑 라인에 둥근 사각형의 함수 아이콘과 함께 간략한 내용으로 표시된다. 함수 매핑이 생성되면 그림과 같이 자동적으로 생성된 함수 매핑이 선택된 상태가 된다.</p>
 (Remove mapping)	표현식 및 매핑을 삭제한다.
 (Paint mapping)	매핑된 필드들을 보라색 블록으로 감싸 하이라이트한 상태로 표시해준다.
 (Remove all mapping)	모든 매핑관계를 제거한다.
Logical	Source/Target의 표시된 정보들을 Logical Name으로 표시한다.
Physical	Source/Target의 표시된 정보들을 Physical Name으로 표시한다.
Add User Class	매핑시키는 로직을 정의해둔 유저 클래스를 검색하는 창을 생성한다.
Add Mapping code	매핑에 대한 사전에 작성한 Java 코드를 추가할 수 있는 창을 생성한다.

매핑 라인은 정의된 매핑이 그림으로 그려지는 영역으로, 매핑은 푸른선으로 표시되고, 현재 선택된 매핑은 붉은 선으로 표시된다. 또한 매핑의 대상 아이템이 소스와 타겟 트리에 나타나는 경우는 실선으로 그려지고, 매핑의 대상이 소스나 타겟 트리에서 접혀있는 부분에 속해 있어 보이지 않는 경우에는 점선으로 그려진다.

정보조회/편집영역

소스 설정(4)과 편집 탭(5) 그리고 속성창(6)으로 구성되어 있다.

• [Location Path]

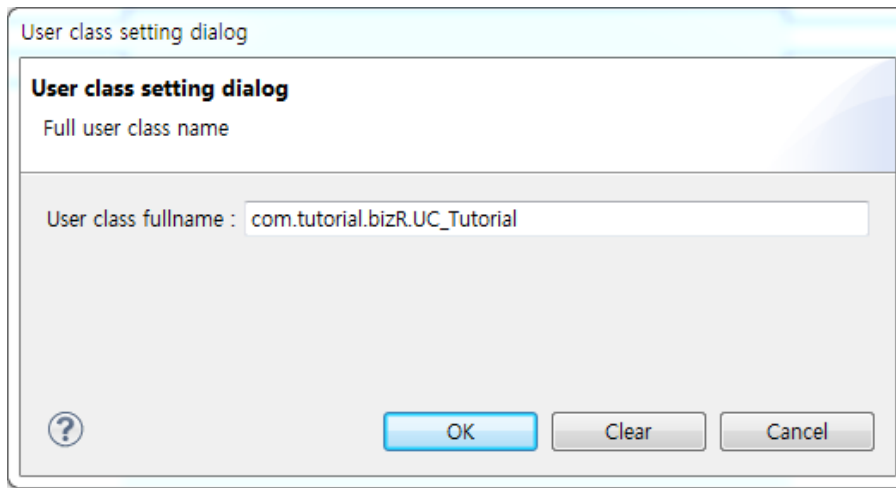
일반적인 매핑의 위치를 보여주는 리스트 탭이다. 편집을 지원하지 않는 읽기전용 화면이며, 이곳에서는 매핑들을 리스트의 형태로 간략하게 보여준다.

• [Expression]

함수 매핑과 상수값을 편집할 수 있는 탭이다.

• [Add User Class]

편집 영역에 **[Add new function]** 메뉴 함수에서 처리할 수 없는 복잡한 로직은 별도의 유저 클래스를 작성하여 등록할 수 있다. **[Add UserClass]** 버튼을 클릭하면 유저 클래스를 등록할 수 있는 대화상자가 다음과 같이 나타난다.



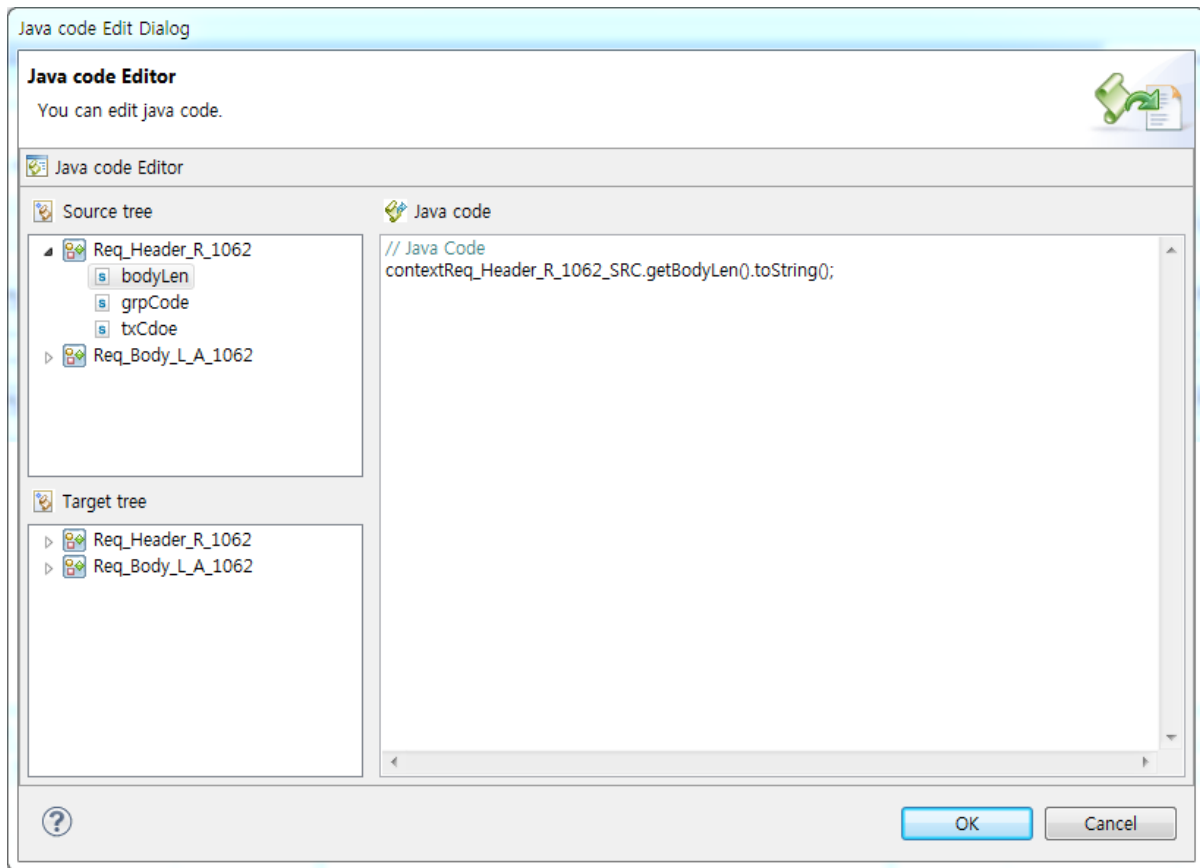
유저 클래스 세팅 다이얼로그

'User class fullname'에 패키지 이름을 포함한 유저 클래스의 전체 이름을 작성한다. 유저 클래스를 서버에 올린 적이 없을 경우 **[Upload]** 버튼을 클릭해서 새로운 라이브러리를 올려야 한다. 이전에 유저 클래스를 올렸을 경우에는 **[OK]** 버튼을 클릭하여 설정을 완료할 수 있다. **[Clear]** 버튼을 클릭하여 유저 클래스 설정을 삭제할 수도 있다. 매핑 기능에는 없지만 단순한 추가 기능이 필요할 경우에는 간단하게 매핑에 추가 코드만 넣어서 처리할 수 있다.

- **[Add Mapping code]**

[Add Mapping code] 버튼을 클릭하면 다음과 같이 Java 코드를 입력할 수 있는 화면이 나타난다.

코드는 매핑이 모두 완료된 이후에 실행하는 코드이다. 변수의 값은 왼쪽의 소스 트리 및 타겟 트리에서 드래그 앤드 드롭으로 가져온다. 컴파일한 결과가 바로 조회되지 않으므로 코드 작성에 유의해야 한다. 타겟 트리 소스 트리와 마찬가지로 타겟 트리의 컨텍스트 메뉴에서 여러 가지 작업을 수행할 수 있다.



Java code 에디터 다이얼로그

다음은 각 영역별 설명이다.



영역	설명
Source Tree(Target Tree)	소스 트리와 타겟 트리는 일반적인 트리 컨트롤과 동작이 동일하다. 마우스를 통해서 선택이 가능하고, 키보드로는 <+>, <-> 키로 트리를 접거나 펼수 있다. 각 소스(타겟) 트리의 아이탬은 아이탬을 상징하는 아이콘과 함께 그려지며, 아이콘에는 객체의 개수 제한 조건이 함께 표시된다.
Java code	소스(타겟)의 속성을 보여주는 영역이다. 노드의 리소스 타입별로 다른 형태의 속성을 나타낼 수 있다.

A.2.2. 매핑 마법사 사용

매핑 마법사는 1개 이상의 소스와 1개 이상의 타겟 사이의 매핑을 지원한다. 그리고 매핑 가능한 소스 아이탬과 타겟 아이탬은 모두 각각의 소스/타겟 트리에 나타나지 않을 수 있다. 소스 트리에 나타나는 아이탬과 타겟 트리에 나타나는 아이탬은 기본적으로 필요한 아이탬들이 나타나고, 사용자가 추가하거나 삭제해서 매핑 대상이 아닌 아이탬을 매핑 마법사에 보이지 않게 할 수 있다.



소스트리

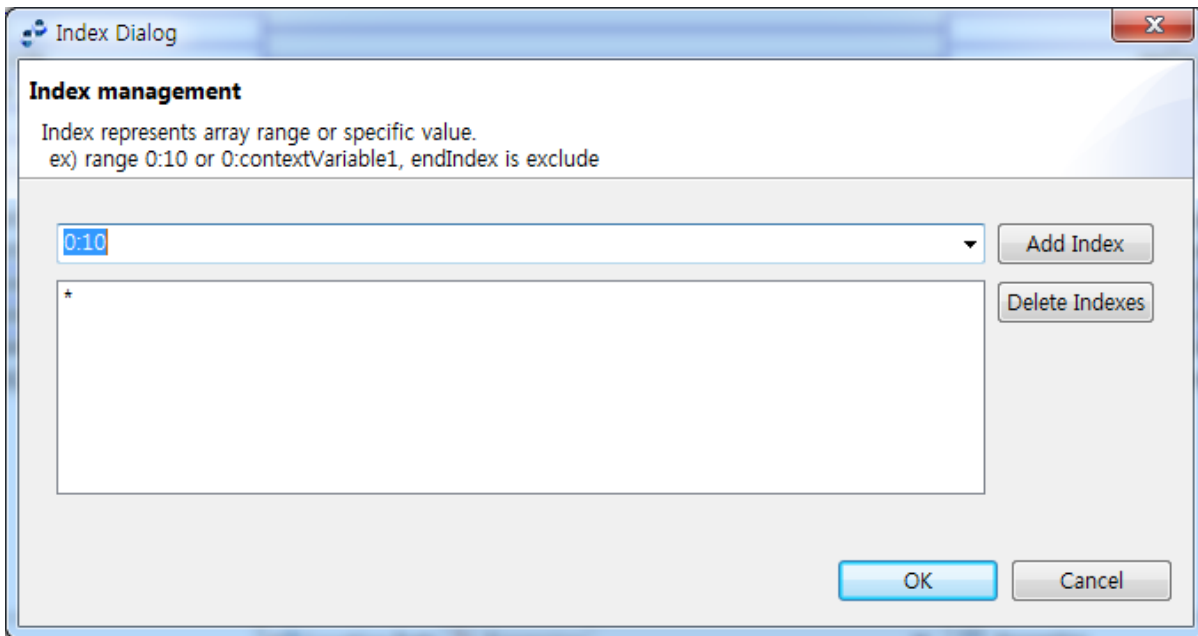
소스 트리의 컨텍스트 메뉴에서 여러 가지 작업을 수행할 수 있다.

Add source
Remove source
Search Node...
Reset visible multiple item
Expand tree
 Generate mappings automatically
 Generate mappings with physical name automatically
Reset layout

소스트리 컨텍스트 메뉴

소스 아이템은 트리에서 자신의 하위 아이템들의 부모가 된다. 소스 아이템을 선택하고 소스 트리에서 컨텍스트 메뉴를 열어 **[Remove Source]**를 선택하거나, **[Delete]**를 클릭하면 리스트에서 소스 트리는 숨겨진다. 소스 트리가 리스트에서 숨겨지면 삭제된 소스 아이템이나 하위 아이템에 대해서 정의된 매핑은 자동으로 삭제된다.

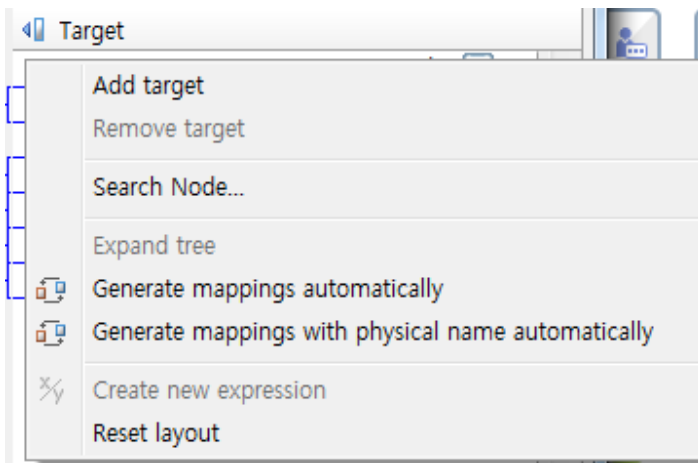
메뉴	설명
[Add source]	소스 트리에 나타나지 않는 아이템들을 선택할 수 있는 Source Selection 대화상자 가 나타난다. Source Selection 대화상자에는 소스 트리에 나타낼 수 있는 모든 아이템이 표시된다. 사용자는 리스트에서 아이템을 클릭하거나 이름을 입력하여 아이템을 선택할 수 있다. <Shift> 키를 누른 상태로 클릭하여 여러 개의 소스를 선택할 수 있다. 선택한 요소를 더블클릭하거나 [OK] 버튼을 클릭하면 선택한 아이템이 소스 트리에 나타난다.
[Remove source]	소스 트리에 나타난 아이템 중 제거할 아이템이 있는 경우 사용한다. 제거 가능한 아이템에 대해서만 활성화 된다.
[Search Node]	Node Search Dialog 에서 트리에서 원하는 노드를 바로 검색할 수 있다. Node Search Dialog에서 ' Search nodes '에 검색할 노드를 입력하면 해당 문자열을 포함한 모든 노드를 검색한다. ' Regular Expression ' 체크박스를 선택하면 단순 문자열 검색이 아닌 정규식을 이용한 검색을 수행한다. 검색이 끝나면 검색된 노드 결과를 알려주는 메시지가 나타나고, 소스 트리에는 검색된 노드들이 선택된 상태로 조회된다.
[Reset visible multiple item]	배열 타입의 아이템인 경우 나타나는 메뉴이며 특정 인덱스에 자유로운 매핑을 보장하기 위해 트리에 나타나는 아이템의 인덱스 범위를 조절할 수 있는 다이얼로그(인덱스 범위 조절 다이얼로그)를 생성한다.
[Expand tree]	선택된 아이템의 하위 아이템을 트리에서 모두 열어서 보여준다.
[Generate mapping automatically]	 아이콘 설명과 동일한 기능을 한다.
[Generate mappings with physical name automatically]	 아이콘 설명과 동일한 기능을 한다.
[Reset layout]	변경된 영역의 크기를 기본 크기로 변환한다. 영역의 크기는 영역 사이의 공간에 마우스 포인터가 변경될 때 드래그하여 변경할 수 있다.



인덱스 범위 조절 다이얼로그

타겟 트리

타겟 트리의 컨텍스트 메뉴에서 여러 가지 작업을 수행할 수 있다.



타겟 트리 컨텍스트 메뉴

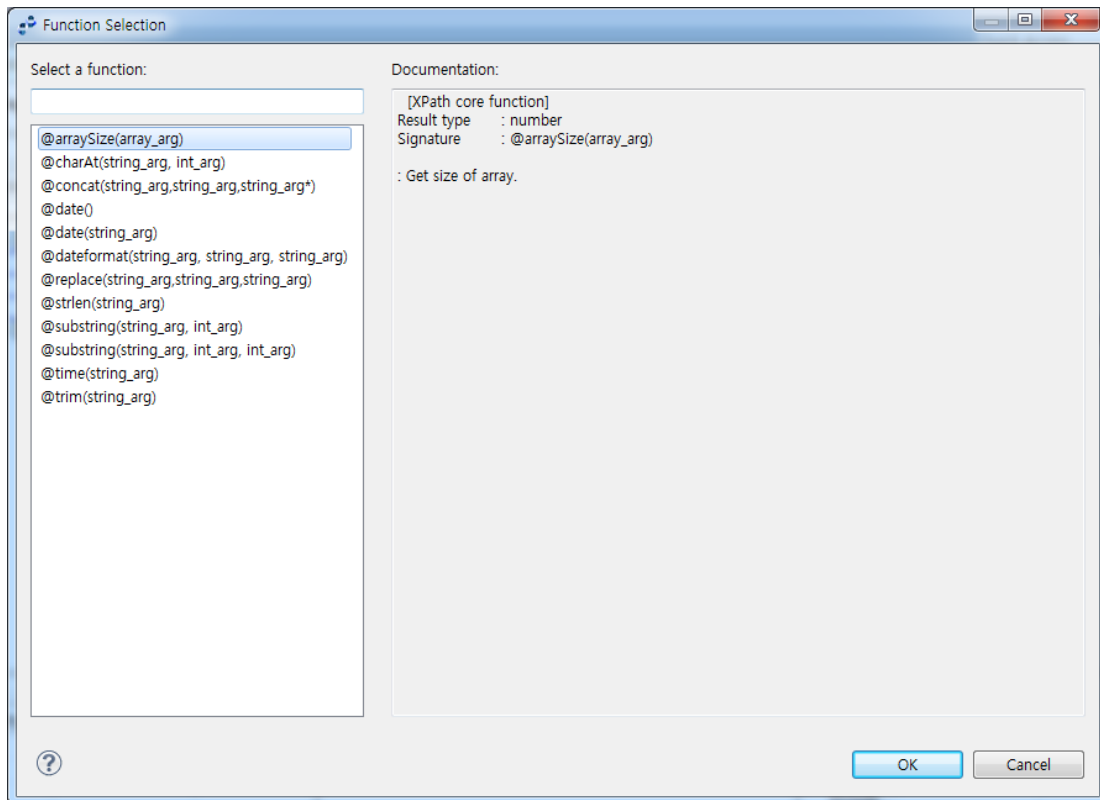
메뉴	설명
[Create new expression]	매핑 관련 영역 상단 bar의  아이콘 설명과 같으므로 생략한다.



타겟 트리의 나머지 컨텍스트 메뉴의 설명은 소스 트리와 동일하다.

편집

함수 매핑이 완료되면 편집 영역에서 해당 함수를 편집해야 한다.

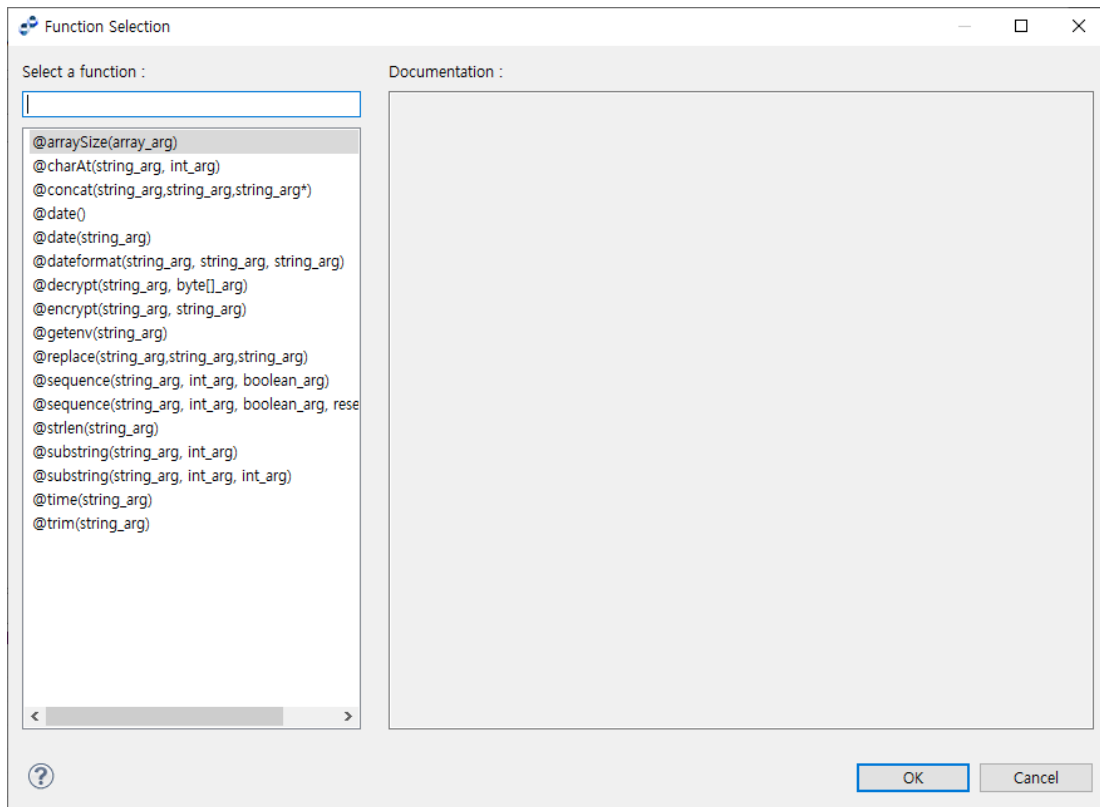


편집 컨텍스트 메뉴

다음은 편집 영역의 컨텍스트 메뉴이다.

- **[Add new Function]**

[Expression] 탭에 새로운 함수를 넣으려면 **[Add new function]** 메뉴를 클릭한다.



함수 선택 메뉴

사용자는 소스 트리에서 직접 함수 편집 창으로 객체를 드래그 앤드 드롭하여 소스 트리에 있는 노드의 값을 가져올 수 있다. 함수의 파라미터를 입력할 때 드래그 앤드 드롭을 이용하면 편리하다.

- ArrayField의 크기를 반환

```
@arraySize(arrayField)
```

- String.charAt(i)

```
@charAt(String, index)
```

- string concatenation

```
@concat(stringField1, stringField2, .....)
```

- 현재의 날짜를 반환(String)

```
@date()
```

- 현재의 날짜를 arg 포맷으로 반환(String)

```
@date("yyyyMMdd");
```

- string dateFormat

```
@dateFormat(stringField, "yyyyMMdd", "yyyy-MM-dd")
@dateFormat(stringField, "HHmmss", "HH:mm:ss")
```

- 문자열을 복호화해서 반환(String)

```
@decrypt(stringField, stringValue);
```

- 문자열을 암호화해서 반환(byte[])

```
@encrypt(stringField, byteArray);
```

- 애니링크에서 제공하는 시스템 변수나 거래 정보들을 반환(String)

```
@getEnv(stringKey);
```

- string replace

```
@replace(stringField, "target", "replacement")
```

- 시퀀스 값을 시스템에 저장하고, 1씩 증가 시키면서 조회한다.

```
@sequence(stringField, int length, boolean padding)
```

- 시퀀스 값을 시스템에 저장하고, 1씩 증가 시키면서 조회한다. resetType으로 값을 reset할지 설정한다.

```
@sequence(stringField, int length, boolean padding, ResetType resetType)
```

- length of string

```
@strlen(stringField)
```

- substring

```
@substring(stringField, int beginIndex)
```

- substring

```
@substring(stringField, int beginIndex, int endIndex)
```

- 현재의 시간을 arg 포맷으로 반환(String)

```
@time("HHmmdd")
```

- string trim

```
@trim(stringField)
```



함수로 해결할 수 없는 복잡한 로직은 **[Add UClass]** 버튼을 클릭해서 유저 클래스를 작성해서 등록할 수 있다.

• [Add variable data]

[Expression] 탭에 소스에 있는 변수값을 가져오려고 하면, 소스의 변수를 **[Expression]** 탭에 드래그하거나 컨텍스트 메뉴에서 **[Add variable data]** 메뉴를 선택한다.

매핑 라인

일반적인 매핑을 정의하기 위해서는 소스 트리과 타겟 트리에서 각각 하나의 아이템을 선택한 후 매핑 영역 상단의 툴바에서 **[Create New Mapping]** 버튼을 클릭하거나 소스 아이템을 드래그하여 타겟 아이템에 올려 놓아도 매핑된다.

소스 아이템과 타겟 아이템은 호환 가능한 타입이고, 타겟에 대하여 어떠한 매핑도 정의되지 않은 상태이어야 한다. 트리에서 선택된 소스 아이템과 타겟 아이템이 호환되지 않고, 이미 정의된 매핑이 있을 경우 툴바의 **[Create New Mapping]** 버튼은 비활성화되며, 소스에서 타겟으로 드래그 앤드 드롭은 허용되지 않는다.

사용자는 매핑 라인을 클릭해서 매핑을 선택할 수 있다. 함수 매핑이나 상수 매핑의 경우 상수 매핑과 함수 매핑을 상징하는 사각형 안쪽을 클릭하여 매핑을 선택할 수 있다. <Ctrl>이나 <Shift> 키를 누른 상태로 클릭하여 2개 이상의 매핑을 선택할 수도 있다.

선택된 매핑이 1개이고 상수 매핑인 경우에 상수 매핑 편집 탭이 활성화되고, 선택된 매핑이 하나이고 함수 매핑인 경우에는 함수 매핑 편집 탭이 활성화된다. 그렇지 않은 경우 매핑 위치 탭이 활성화되어 선택된 매핑 위치의 정보를 보여준다.

트리 속성

속성 창은 트리에서 선택된 아이템의 속성을 보여준다. 소스 속성 창은 소스 트리의 선택된 아이템을 타겟 속성 창은 타겟 트리에서 선택된 아이템의 속성을 각각 보여준다.

다음은 트리에서 선택된 아이템의 속성을 조회하는 화면이다.

종별코드		
Property	Value	
Logical ...	종별코드	
Physical...	kind_code	
Field Type	string	
Length	4	
Include ...		
Array	false	
Decimal	0	

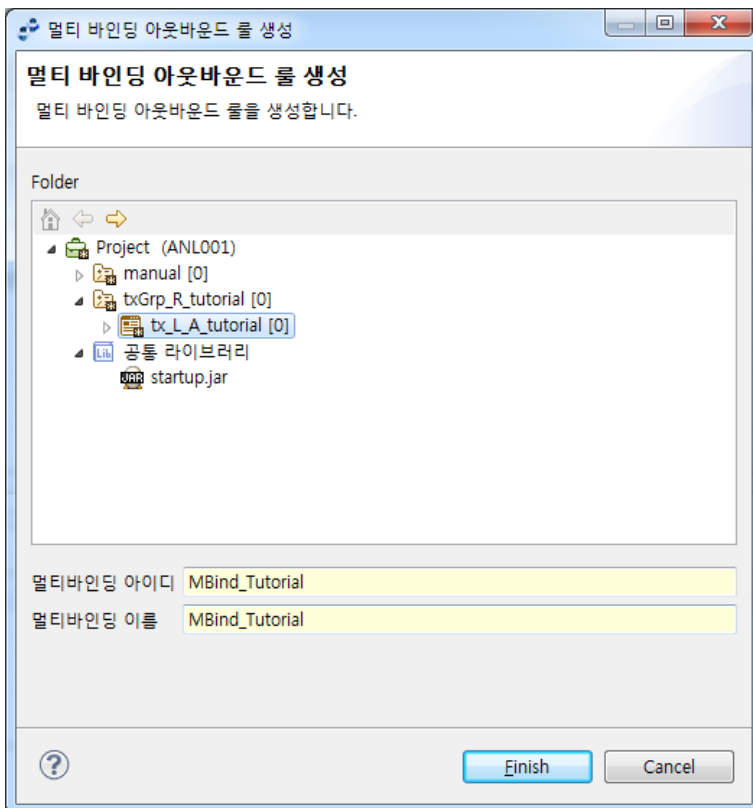
아이템 속성

A.3. 리소스 관련 다이얼로그

본 절에서는 리소스 선택과 관련된 항목에서 볼 수 있는 다이얼로그들에 대해 설명한다.

A.3.1. 리소스 생성 다이얼로그

리소스 생성 다이얼로그는 스튜디오 프로젝트 내에 리소스를 생성해야 할 경우 나타난다. 원하는 거래/거래그룹 항목을 선택한 뒤 **[리소스 이름]**과 **[리소스 ID]**를 설정하고 **[Finish]** 버튼을 클릭하면 선택한 위치 하위 항목으로 리소스가 추가된다. 리소스 마다 선택할 수 있는 옵션과, ID/이름을 지정할 수 있는 방법이 다르므로 리소스를 추가하는 항목의 내용을 잘 살펴보도록 한다.



리소스 생성 다이얼로그

