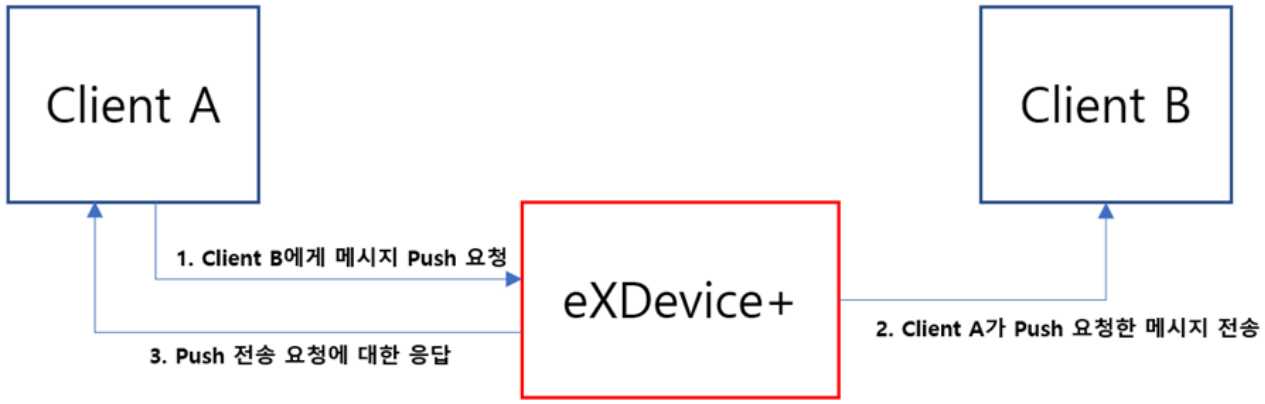
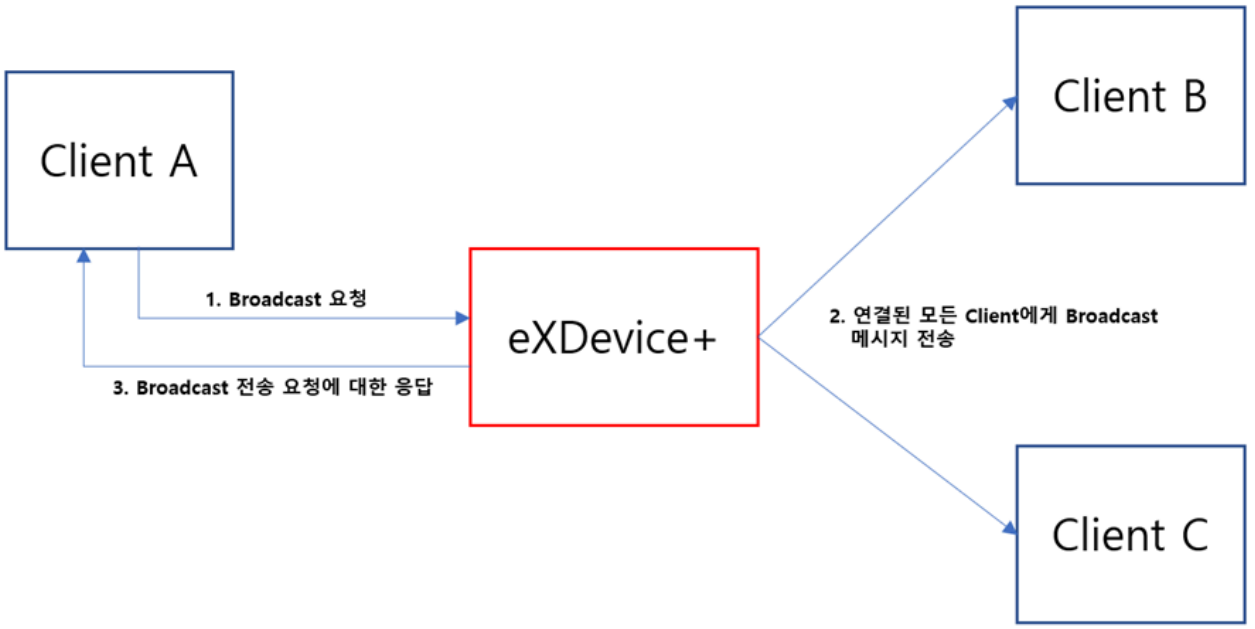
WebSocket client간 Websocket 통신 서비스

현재 PC에서 구동 중인 Websocket client(브라우저, exe 등)들이 eXDevice+에 Websocket으로 연결하여 서로 Websocket 통신을 수행하기 위한 기능들을 제공합니다.

eXDevice+를 사용한 client간 메시지를 송수하는 프로세스는 다음과 같습니다.



<WebSocket client간 Message Push>



<WebSocket client간 Message Broadcast>

서비스 리스트

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

WebSocket client간 메시지를 주고 받는 서비스 리스트는 다음과 같습니다.

No.	서비스명	API명	비고
1	Client 등록 및 수정	UpdateWsInfo	
2	Client 리스트 획득	GetWsClients	
3	자신의 식별 정보 조회	GetWsOwnInfo	
4	push	PushWsMessage	- Unicast, Multicast 지원 - ReceiveWsPush로 메시지 수신
5	broadcast	BroadcastWsMessage	ReceiveWsBroadcast로 메시지 수신
6	push 수신	ReceiveWsPush	직접 호출 불가
7	broadcast 수신	ReceiveWsBroadcast	직접 호출 불가

서비스 상태 코드 리스트

본 서비스에서 공통으로 사용되는 서비스 상태코드 리스트에 대해 정의합니다.

이 상태 코드는 응답 JSON 메시지 중 *return* 또는 *returnValue*의 값에 해당합니다.

리턴코드	리턴 코드 내용	대상 서비스	비고
-1	정의되지 않은 예외	공통	Log 파일에서 내용 확인 필요
0	정상	공통	-
1	PUSH 메시지를 수신할 client가 존재하지 않음	PushWsMessage	-
2	일부 client에게 메시지 송신 실패	PushWsMessage, BroadcastWsMessage	-
3	자신 외 다른 client가 연결되어 있지 않음	BroadcastWsMessage	-
4	등록 요청한 식별 정보가 이미 등록되어 있음	UpdateWsInfo	-
5	등록할 수 없는 식별 정보 데이터	UpdateWsInfo	-

1. Client 등록 및 수정

API명

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

- UpdateWsInfo

정의

- 서비스를 호출한 WebSocket client의 식별 정보(Info)를 갱신합니다.

설명

- eXDevice+는 WebSocket client가 연결되면 해당 client를 식별하기 위한 ID(GUID)를 자동으로 발급 및 관리합니다.
- 본 서비스를 사용해 다른 client가 자신을 식별 할 수 있도록 식별 정보를 등록 또는 갱신 할 수 있습니다.

> ⚠ Client의 식별 정보(Info)는 고유한 정보이므로, 이미 다른 client가 등록한 식별 정보가 있는 경우 다른 client는 동일한 값의 식별 정보를 등록할 수 없습니다.

호출 예시

```
{
  "service": "UpdateWsInfo",
  "requestKey": "Random Request Key",
  "param": {
    "info": "Client를 설명할 수 있는 정보"
  }
}
```

응답 예시

```
{
  "service":"UpdateWsInfo",
  "requestKey":"호출시 사용한 Request Key",
  "statusCode":"0000",
  "return":0
}
```

- return

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

- 서비스 상태 코드.

2. Client 리스트 획득

API명

- GetWsClients

정의

- eXDevice+에 연결된 WebSocket client 들의 ID(GUID) 및 식별 정보(Info) 리스트를 획득합니다.

> `UpdateWsInfo`로 식별 정보를 등록한 client의 경우 식별 정보 값을 획득 할 수 있으며, 식별 정보를 등록하지 않은 client는 빈 값("")이 리턴 됩니다.

호출 예시

```
{
  "service": "GetWsClients",
  "requestKey": "Random Request Key",
  "param":{}
}
```

응답 예시

```
{
  "service": "GetWsClients",
  "requestKey": "호출시 사용한 Request Key",
  "statusCode": "0000",
  "return":{
    "returnValue": 0,
    "infos": [
      {"id": "aaa-bbb-ccc", "info": "eXDevice+ Sample"},
      {"id": "000-111-222", "info": ""}
    ]
```

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

```
  }
}
```

- returnValue
  - 서비스 상태 코드.

- infos: eXDevice+에 연결된 WebSocket client ID 및 식별 정보.
  - id
    - WebSocket client 고유 ID(GUID).

  - info
    - WebSocket client 식별 정보.

3. 자신의 식별 정보 조회

API명

- GetWsOwnInfo

정의

- eXDevice+에 등록된 WebSocket client 자신의 ID(GUID) 및 등록한 식별 정보(Info)를 획득합니다.

> `UpdateWsInfo`로 식별 정보를 등록한 client의 경우 식별 정보 값을 획득 할 수 있으며, 식별 정보를 등록하지 않은 client는 빈 값("")이 리턴 됩니다.

호출 예시

```
{
  "service": "GetWsOwnInfo",
  "requestKey": "Random Request Key",
```

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

```
  "param": {}
}
```

응답 예시

```
{
  "service": "GetWsOwnInfo",
  "requestKey": "호출시 사용한 Request Key",
  "statusCode": "0000",
  "return":{
    "returnValue": 0,
    "id": "aaa-bbb-ccc",
    "info": "eXDevice+ Sample"
  }
}
```

- returnValue
  - 서비스 상태 코드.
- id
  - WebSocket client 고유 ID(GUID).
- info
  - WebSocket client 식별 정보.

4. push

API명

- PushWsMessage

정의

- eXDevice+에 연결된 WebSocket client에게 WebSocket 메시지를 송신하기 위해 사용됩니다.
- 본 서비스는 단일 Websocket client에게 메시지를 송신하는 **Unicast** 방식, 다수의 WebSocket client에게 메시지를 송신하는 **Multicast** 방식을 지원합니다.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

Push 메시지를 수신 받는 WebSocket client의 수신 프로토콜은 `ReceiveWsPush`를 참고하시길 바랍니다.

송신 상태 코드 리스트

- Push와 Broadcast는 서비스 상태 코드 외 메시지 송신 상태 코드를 가집니다.
- 송신 상태 코드는 *returnValue*에 0 외의 값이 들어오는 경우 *status*에 다음과 같은 송신 상태 코드가 리턴됩니다.

상태코드	상태 코드 내용	비고
1	메시지 수신 대상 WebSocket client 연결이 해제됨	
2	대상 WebSocket client에게 메시지 송신 실패	
3	서비스 요청한 WebSocket client 외 다른 WebSocket client가 연결되어 있지 않음	
4	대상 WebSocket client가 eXDevice+에 연결되어 있지 않음	

호출 예시

1. Unicast

Unicast는 하나의 WebSocket client에게 Websocket 메시지를 송신하는 방식입니다. Unicast로 메시지 송신 시 **"ID"**, **"식별 정보"**, **"ID&식별 정보"**를 사용해 메시지를 송신할 WebSocket client를 지정할 수 있습니다.

1) ID(GUID)를 사용한 Unicast

```
{
  "service": "PushWsMessage",
  "requestKey": "Random Request Key",
  "param":{
    "receiver": "aaa-bbb-ccc",
    "message": "message to unicast"
  }
}
```

- receiver

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

    - 메시지를 수신 받을 WebSocket client의 ID.

- message
    - 송신할 메시지.

2) 식별 정보를 사용한 Unicast

```
{
  "service": "PushWsMessage",
  "requestKey": "Random Request Key",
  "param":{
    "receiver":{
      "info": "Test WebSocket Client"
    },
    "message": "message to unicast"
  }
}
```

- receiver: 메시지를 수신 받을 WebSocket client의 정보.
    - info
        - WebSocket client 식별 정보.

- message
    - 송신할 메시지.

3) ID와 식별 정보 모두를 사용한 Unicast

```
{
  "service": "PushWsMessage",
  "requestKey": "Random Request Key",
  "param":{
  "receiver":{
    "id" :"aaa-bbb-ccc",
    "info": "Test Websocket Client"
  },
  "message": "message to unicast"
```

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

```
  }
}
```

- receiver: 메시지를 수신 받을 WebSocket client의 상세 정보.
  - id
    - WebSocket client의 ID.
  - info
    - WebSocket client의 식별 정보.

- message
  - 송신할 메시지.

2. Multicast

Multicast는 다수의 WebSocket client에게 Websocket 메시지를 송신하는 방식입니다. Multicast로 메시지 송신 시 **"ID"**, **"식별 정보"**, **"ID&식별 정보"**를 사용해 메시지를 송신할 WebSocket client를 지정할 수 있습니다.

1) ID(GUID)를 사용한 Multicast

```
{
  "service": "PushWsMessage",
  "requestKey": "Random Request Key",
  "param": {
    "receiver": ["aaa-bbb-ccc", "000-111-222"],
    "message": "message to multicast"
  }
}
```

- receiver
  - 메시지를 수신 받을 WebSocket client의 ID 배열.

- message
  - 송신할 메시지.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

2) 식별 정보를 사용한 Multicast

```
{
  "service": "PushWsMessage",
  "requestKey": "Random Request Key",
  "param": {
    "receiver": [
      {"info": "Test Client1"},
      {"info": "Test Client2"}
    ],
    "message" :"message to multicast"
  }
}
```

- receiver: 메시지를 수신 받을 WebSocket client의 정보 배열.
  - info
    - WebSocket client 식별 정보.

- message
  - 송신할 메시지.

3) ID와 식별 정보 모두를 사용한 Multicast

```
{
  "service": "PushWsMessage",
  "requestKey": "Random Request Key",
  "param": {
    "receiver": [
      {"id": "aaa-bbb-ccc", "info": "Test Client1"},
      {"id": "000-111-222", "info": "Test Client2"}
    ],
    "message": "message to multicast"
  }
}
```

- receiver: 메시지를 수신 받을 WebSocket client들의 상세 정보 배열.
  - id
    - WebSocket client의 ID.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

    - info
        - WebSocket client의 식별 정보.

- message
    - 송신할 메시지.

응답 예시

1. Push 성공

Unicast, Multicast시 메시지 송신에 성공하면 eXDevice+로부터 다음과 같은 응답 메시지를 수신 받습니다.

> Push 실패 판단은 *returnValue*의 값에 따라 판단합니다.

```
{
  "service": "PushWsMessage",
  "requestKey": "호출시 사용한 Request Key",
  "statusCode": "0000",
  "return":{
    "returnValue":0,
    "clients":[]
  }
}
```

- returnValue
    - 서비스 상태 코드.

- clients
    - 송신 실패한 WebSocket client 정보 및 송신 실패 상태 코드.
    - 모든 client에 송신 성공 한 경우 빈 배열 리턴.

2. Push 실패

Unicast, Multicast시 메시지 송신에 실패하면 *returnValue*로 0 외의 값을 리턴 받습니다.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

각 client별 송신 실패 원인은 *status*의 송신 상태 코드를 통해 확인할 수 있습니다.

```
{
  "service": "PushWsMessage",
  "requestKey": "호출시 사용한 Request Key",
  "statusCode": "0000",
  "return":{
    "returnValue":2,
    "clients":[
      {"id": "aaa-bbb-ccc", "info": "Test WebSocket Client", "status":1}
    ]
  }
}
```

- returnValue
  - 서비스 상태 코드

- clients: 송신 실패한 WebSocket client 정보.
  - id
    - WebSocket client의 ID.
  - info
    - WebSocket client의 식별 정보.
  - status
    - 송신 상태 코드.

5. broadcast

API명

- BroadcastWsMessage

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

정의

- eXDevice+에 연결된 모든 WebSocket client에게 WebSocket 메시지를 송신하기 위해 사용됩니다.

> Broadcast 메시지를 수신 받는 WebSocket client의 수신 프로토콜은 `ReceiveWsBroadcast`를 참고하시길 바랍니다.

송신 상태 코드 리스트

- Push와 Broadcast는 메시지 송신 시 서비스 상태 코드 외 메시지 송신 상태 코드를 가집니다.
- 송신 상태 코드는 *returnValue*에 0 외의 값이 들어오는 경우, 메시지를 수신 받는 WebSocket client의 정보 중 *status*에 다음과 같은 송신 상태 코드가 리턴됩니다.

상태코드	상태 코드 내용	비고
1	메시지 수신 대상 WebSocket client 연결이 해제됨	
2	대상 WebSocket client에게 메시지 송신 실패	
3	서비스 요청한 WebSocket client 외 다른 WebSocket client가 연결되어 있지 않음	
4	대상 WebSocket client가 eXDevice+에 연결되어 있지 않음	

호출 예시

```
{
  "service": "BroadcastWsMessage",
  "requestKey": "Random Request Key",
  "param": {
    "mode":1
    "message": "message to broadcast"
  }
}
```

- mode: Broadcast 메시지 송신 방식.
  - 1: 서비스를 호출하는 WebSocket client를 제외한 eXDevice+에 연결된 모든 client에게 Broadcast.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

  - 2: 서비스를 호출하는 WebSocket client를 포함한 eXDevice+에 연결된 모든 client에게 Broadcast.

- message
  - 송신할 메시지.

응답 예시

1. Broadcast 성공

Broadcast시 메시지 송신에 성공하면 eXDevice+로부터 다음과 같은 응답 메시지를 수신 받습니다.

> Broadcast 실패 판단은 *returnValue*의 값에 따라 판단합니다.

```
{
  "service": "BroadcastWsMessage",
  "requestKey": "호출시 사용한 Request Key",
  "statusCode": "0000",
  "return":{
    "returnValue":0,
    "clients":[]
  }
}
```

- returnValue
  - 서비스 상태 코드.

- clients
  - 송신 실패한 WebSocket client 정보 및 송신 실패 상태 코드.
  - 모든 client에 송신 성공 한 경우 빈 배열 리턴.

2. Broadcast 실패

Broadcast시 메시지 송신에 실패하면 *returnValue*로 0 외의 값을 리턴 받습니다.

> 각 client별 송신 실패 원인은 *status*의 송신 상태 코드를 통해 확인할 수 있습니다.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

```
{
  "service": "BroadcastWsMessage",
  "requestKey": "호출시 사용한 Request Key",
  "statusCode": "0000",
  "return":{
    "returnValue":2,
    "clients":[
      {"id": "aaa-bbb-ccc", "info": "Test WebSocket Client", "status":1}
    ]
  }
}
```

- returnValue
  - 서비스 상태코드.

- clients: 송신 실패한 WebSocket client 정보.
  - id
    - WebSocket client의 ID.
  - info
    - WebSocket client의 식별 정보.
  - status
    - 송신 상태 코드.

6. push 수신

API명

- ReceiveWsPush

정의

- 다른 WebSocket client가 push한 메시지를 수신합니다.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

설명

- `PushWsMessage`로 다른 WebSocket client가 자신에게 송신한 메시지를 본 서비스를 통해 수신 받습니다.
- 본 서비스로 Push 메시지를 수신 받으려면 WebSocket 이벤트 중 *onmessage* 이벤트를 통해 수신합니다.

> ⚠ 본 서비스는 직접 호출할 수 없습니다.

응답 예시

```
{
  "service":"ReceiveWsPush",
  "sender":{
    "id":"aaa-bbb-ccc",
    "info":"Sender test websocket"
  },
  "message":"Push message test."
}
```

- sender: Push 메시지를 송신한 WebSocket client의 정보.
  - id
    - Push 메시지를 송신한 client의 ID.
  - info
    - Push 메시지를 송신한 client의 식별 정보.
- message
  - 수신 받은 메시지.

7. broadcast 수신

API명

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

- ReceiveWsBroadcast

정의

- 다른 WebSocket client가 broadcast한 메시지를 수신합니다.

설명

- `BroadcastWsMessage`로 다른 WebSocket client가 자신에게 송신한 메시지를 본 서비스를 통해 수신 받습니다.
- 본 서비스로 Broadcast 메시지를 수신 받으려면 WebSocket 이벤트 중 *onmessage* 이벤트를 통해 수신합니다.

> ⚠ 본 서비스는 직접 호출할 수 없습니다.

응답 예시

```
{
  "service":"ReceiveWsBroadcast",
  "sender":{
    "id":"aaa-bbb-ccc",
    "info":"Sender test websocket"
  },
  "message":"Push message test."
}
```

- sender: Broadcast 메시지를 송신한 WebSocket client의 정보.
  - id
    - Broadcast 메시지를 송신한 client의 ID.
  - info
    - Broadcast 메시지를 송신한 client의 식별 정보.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.

- message
  - 수신 받은 메시지.

Copyright ⓒ TOMATO SYSTEM CO., LTD. All rights reserved.