“Dynamic Push Message” 프로젝트

튜토리얼

3100-01

Ver 1.5

JOYTUNE 프로젝트팀

개 정 이 력

버전	작성일	변경내용[1]	작성자	승인자
1.0	2014.11.11	최초 작성	오재호	
1.1	2014.11.25	BUSINESS SERVER API 추가	오재호	
1.2	2014.12.11	인증/보안 추가	오재호	
1.3	2014.12.19	QUICK START GUIDE 추가	오재호	
1.4	2014.02.06	잘못된 문구 수정	오재호	
1.5	2014.03.13	PRODUCTKEY 설정 추가	오재호	

[1] 변경 내용: 변경이 발생되는 위치와 변경 내용을 자세히 기록(장/절과 변경 내용을 기술한다.)

목 차

1. Quick Start Guide

1.1 JavaScript

1.1.1 PRODUCTKEY 발급받기

www.dpush.co.kr 에서 PRODUCTKEY 를 발급받습니다.
간단한 가입을 통한 무료 PRODUCTKEY 가 발급됩니다.

1.1.2 javascript library include

DPMessage 의 javascript library 를 include 합니다.

```
<script src="http://jslib.dpush.co.kr:9000"></script>
```

1.1.3 DPClient 선언

DPClient 를 선언하면서 DPMessage 서버와의 connection 이 생성됩니다.

```
var client = new DPClient('[발급받은 PRODUCTKEY]');
```

1.1.4 그룹 생성

메시지를 전달받을 그룹에 접속(최초는 생성)합니다.

```
var group = client.openGroup('test-group');
```

1.1.5 Action OnReceive 함수 정의

메시지를 받을 onReceive 함수를 정의합니다.
*onReceive 는 action 이라는 작업 단위로 구분됩니다.

```
group.onReceive('test-action', function(data, userid, custinfo) {
   alert(data);
});
```

1.1.6 서버 API 를 통한 메시지 전달

Client 에서 직접 메시지를 보내는 것도 가능합니다.
지금은 SERVER API (BUSINESS SERVER API)를 통해서 메시지를 전달하는 방법을 PHP 를 기준으로 가이드합니다.
download 에서 DPServerPHP.zip 파일을 풀어서 서버에 업로드 하고 dpmessage.php 를 include 합니다.

```
include "dpmessage.php";
```

메시지를 전송합니다.

```
$dpmessage = new DPServer('[발급받은 PRODUCTKEY]');
$dpmessage->send('test-group','test-action',"Hello World!!");
```

1.2 Android

1.2.1 PRODUCTKEY 발급받기

www.dpush.co.kr 에서 PRODUCTKEY 를 발급받습니다.
간단한 가입을 통한 무료 PRODUCTKEY 가 발급됩니다.

1.2.2 JAVA library(JAR File)

download 에서 dpmessage.jar, Java-WebSocket-x.x.x.jar, json-xxxxxxxx.jar 파일을 다운받아서 패키지에 등록합니다.
Java-WebSocket 과 json 의 경우 기존의 사용하시는 libary 버전에 대한 테스트를 수행하시고 작업하시면 됩니다.

1.2.3 JAVA library import

해당 라이브러리 파일을 import 합니다.

```
import kr.co.dpush.client.DPClient;
import kr.co.dpush.client.GroupInfo;
import kr.co.dpush.client.Callback;
```

1.2.4 DPClient 선언

DPClient 를 선언하면서 DPMessage 서버와의 connection 이 생성됩니다.

```
DPClient client = new DPClient('[발급받은 PRODUCTKEY]');
```

1.2.5 그룹 생성

메시지를 전달받을 그룹에 접속(최초는 생성)합니다.

```
GroupInfo group = client.openGroup('test-group');
```

1.2.6 Action OnReceive 함수 정의

메시지를 받을 onReceive 함수를 정의합니다.
*onReceive 는 action 이라는 작업 단위로 구분됩니다.

```
group.onReceive('test-action', new Callback() {
        @Override
        public void call(Object... args) {
           System.out.println(args[0]);
```

```
        }});
```

1.2.7 서버 API 를 통한 메시지 전달

Client 에서 직접 메시지를 보내는 것도 가능합니다.
지금은 SERVER API (BUSINESS SERVER API)를 통해서 메시지를 전달하는 방법을 PHP 를 기준으로 가이드합니다.
download 에서 DPServerPHP.zip 파일을 풀어서 서버에 업로드 하고 dpmessage.php 를 include 합니다.

```
include "dpmessage.php";
```

메시지를 전송합니다.

```
$dpmessage = new DPServer('[발급받은 PRODUCTKEY]');
$dpmessage->send('test-group','test-action',"Hello World!!");
```

2. Client 튜토리얼

2.1 DPClient 선언

* Client 튜토리얼은 javascript 를 기준으로 설명합니다. android, ios 도 같은 함수명으로 API 를 참조하시면 됩니다.
DPClient 를 선언을 통해 각 Client 들은 DPush 서버에 connection 을 하게됩니다.

먼저 javascript library 를 include 합니다.
<script src="http://localhost:9000"></script>

기본선언은 다음과 같습니다.
var client = new DPClient('발급받은 PRODUCT KEY');

```
var client = new DPClient(productkey, options);
```

- productkey (String) required
  발급받은 Product Key
- options (Object) optional
  DPMessage 옵션정보
  - security (Boolean) : SSL 보안을 사용할지의 유무. 기본값 false
  - resultcallback (Function) : debug 용 callback 함수

  SSL 보안의 경우 BASIC 에서는 지원하지 않습니다. 관련 내용은 5.1 장을 참조하시면 됩니다.

sample)

```
var client = new DPClient('TESTPRODUCTKEY', {
      'security' : false,
      'resultcallback' : resultcallback
   }
);
```

2.2 Group Open

그룹은 Push 대상의 집합을 만드는 과정입니다. 지정된 그룹에 속한 유저에게 메시지를 Broadcast 로 Push 하게 됩니다.

기본선언은 다음과 같습니다.
var group = client.openGroup('그룹명');

```
var group = client.openGroup (groupname, options, callback);
```

- groupname (String) required
  임의의 그룹명
- options (Object) optional
  DPMessage 옵션정보
  - custevent (Boolean) : 그룹에서 CUSTMER INFO 이벤트를 callback 받는 유무. 기본값 false
  - custinfo (Object) : 연결된 그룹의 CUSTOMER 정보

- sendevent (Boolean) : 해당 그룹의 client 가 send 함수를 사용가능한 유무. 기본값 false
- authparam (String) : 인증 URL 전달 파라미터
- authcallback (Function) : 인증 후 실행될 callback 함수

CUSTOMER INFO 를 이용하게 되면 해당 client 만의 고유한 정보를 전체 그룹에서 공유하게 됩니다. 가령 채팅방의 닉네임같은 정보입니다. CUSTOMER INFO 를 사용할 경우 client 의 접속, 탈퇴, 정보변경 시에 메시지를 보내기 때문에 메시지 count 가 많이 증가하게 됩니다. CUSTOMER 정보관련 내용은 2 장을 참조하시면 됩니다. CUSTOMER INFO 를 사용하기 위해서는 PRODUCTKEY 설정에서 Customer Info 사용을 체크하셔야 합니다.
sendevent 는 client 에서 메시지를 보낼 수 있는 send 함수의 사용 유무입니다. 채팅의 경우 기본적으로 true 로 세팅하셔야 합니다. client 의 send 를 사용하기 위해서는 PRODUCTKEY 설정에서 클라이언트 메세지를 체크하셔야 합니다.
authparam, authcallback 등 인증관련 내용은 5.2 장을 참조하시면 됩니다.

- callback (Function) optional
  그룹이 생성된 후의 이벤트가 필요할 경우 선언

sample)

```
var group = client.openGroup('chat-group', {
      'custevent' : true,
      'custinfo' : {'nickname' : '홍길동'},
      'sendevent' : true,
      'authparam' : "param1=param1&param2=param2",
      'authcallback' : authcallback
   }, function(data) {
      alert(data);
});
```

2.3 OnReceive & Send

메시지 수신과 송신은 ActionName 으로 구분됩니다. Action 은 Push 되는 그룹에서 해당 데이터의 행위를 정의하는 내용입니다.
Action 을 OnReceive 하면 해당 Action 에 대한 Push Message 가 해당 Action 의 callback 함수로 전달됩니다.
기본선언은 다음과 같습니다.
group.onReceive('액션명', callback 함수);

```
group.onReceive (actionname, callback);
```

- actionname (String) required
  임의의 액션명
- callback (Function) requried
  메시지를 전달 받는 callback 함수
  callback function parameter
  - data (String) : 전달된 Push 메시지
  - userid (String) : 접속한 그룹의 개인 Qnique 한 ID 값 (시스템값)
  - custinfo (Object) : Push 메시지를 보낸 그룹의 개인 정보

sample)

```
group.onReceive('chat-action', function(data, userid, custinfo) {
    alert(data);
    alert(userid);
    alert(custinfo);
});
```

Send Action 은 소속되어 있는 Group 에 해당 메시지를 보내는 행위입니다. Client 에서 Send Action 을 사용하고 싶은 경우 먼저 "마이페이지"의 해당 "PRODUCTKEY 설정"에서 "클라이언트 메시지" 의 "전송가능"을 체크하셔야 하고 Client 소스 코딩에서 openGroup 시에 sendevent 를 true 로 정의하셔야 합니다. 기본은 false 로 정의되어 있습니다.
Action 명, 데이터 내용을 파라미터로 Push 를 하면 Group 에 소속되어 있는 Client 에게 Push 메시지가 전달되고 해당 Client 중에서 Action 명으로 onReceive 되어 있는 함수로 데이터가 전달 됩니다.
기본선언은 다음과 같습니다.
group.send('액션명', '데이터');

```
group.send (actionname, pushdata);
```

- actionname (String) required
  임의의 액션명
- pushdata (String) required
  전달할 데이터의 내용

sample)

```
group.send('chat-action', 'Hello World!');
```

2.4 Debug For Client API

Client API 의 기본 Debug 는 alert 혹은 console.log 를 통해서 확인 할 수 있습니다. 각각의 API 를 호출 시 DPMessage 서버와의 통신이 발생합니다. 각 통신 후에 결과를 확인하기 위해서는 DPClient 선언 시 options 에 resultcallback 을 지원하도록 되어 있습니다. resultcallback function 에 결과코드(resultcd), 상세(resultdesc) 의 파라미터를 같이 넘기며 해당 내용을 통해서 통신 결과를 알 수 있습니다. 결과코드 0 의 경우 성공이며 그 외의 경우는 참고의 에러코드를 확인하시면 됩니다. 다음의 간단한 예제를 통해서 확인할 수 있습니다.

sample)

```
var resultcallback = function(resultcode, resultdesc) {
    if(resultcode == 0) {
        console.log("통신 결과 성공");
    } else {
        console.log("통신 에러 : " + resultdesc + " (" + resultcode + ")";
    }
}
client = new DPClient('TESTCHAT', {
        'resultcallback' : resultcallback
    }, function(data) {
        writeUsers();
```

```
    });
```

2.5 간단한 채팅 예제

지금까지의 내용을 바탕으로 간단한 채팅을 구성해 보도록 하겠습니다.

html 의 body 안에 다음과 같은 간단한 챗창을 만듭니다.

```
<form name="chatform" onsubmit="return false;">
<div style="width:1020px; height:500px;">
<div id="chatscreen" style="width:90%;height:90%;border:1px solid green;overflow:auto; float:left"></div>
<input type="text" name="chatinput" id="chatinput" style="width:80%" onKeyPress="inputTalk()"/>
<input type="button" value=">>" onclick="talk()" style="width:10%">
</div>
</form>
```

다음의 javascript 라이브러리를 include 합니다.

```
<script src="http://jslib.dpush.co.kr:9000"></script>
```

<script> 구문 안에 다음의 javascript 를 코딩합니다.
client 의 메시지 송신은 openGroup 선언 시에 sendevent 를 true 값으로 세팅하셔야 합니다.

```
var client = new DPClient('TESTCHAT'); // DPClient 생성
var group = client.openGroup('chat-group', {'sendevent' : true}); // group open
group.onReceive('chat-action', function(data, userid, custinfo) {  //  onReceive action
    var chatscreen = document.getElementById("chatscreen");
    chatscreen.innerHTML += data + "<br>";
    chatscreen.scrollTop = chatscreen.scrollHeight;
});
function talk() {
    var txt = document.getElementById("chatinput").value;
    document.getElementById("chatinput").value = '';
    group.send('chat-action', txt);  // send action
}
```

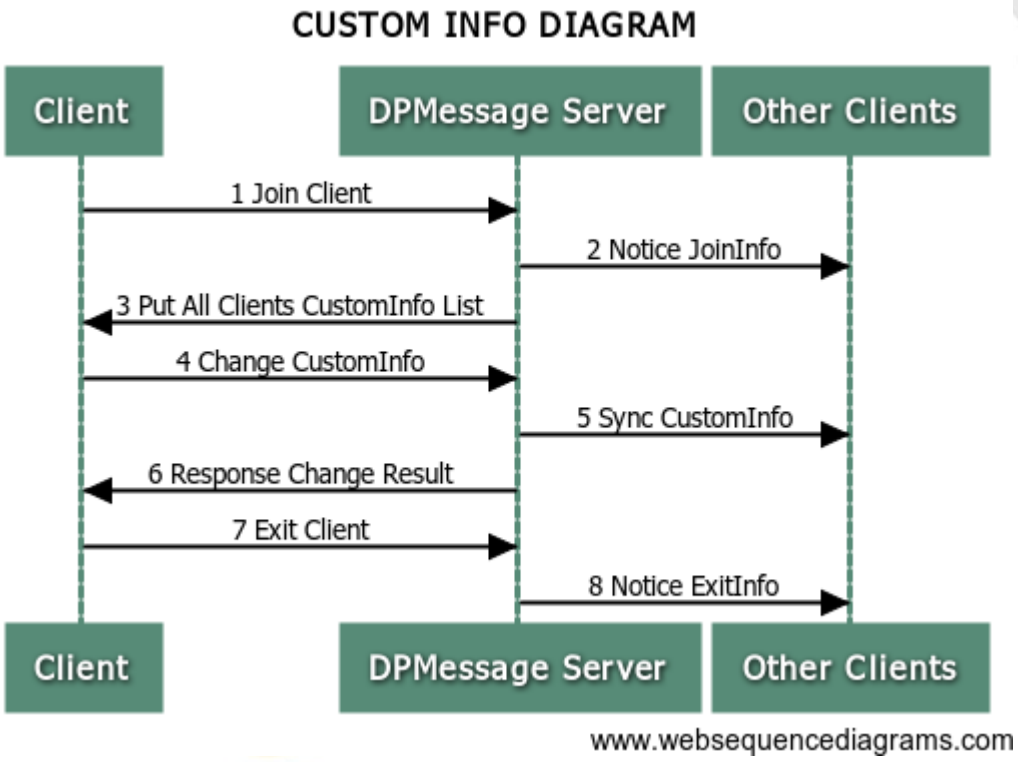
실제 간단한 채팅에 필요한 javascript 는 얼마 되지 않습니다. 해당 풀소스는 simplechat.html 을 참고하시면 됩니다. 다음 챕터에서는 CUSTOMER info 에 대한 자세한 내용을 다루겠습니다. 샘플로 현재 챗창에서 nickname 과 나이 성별을 입력 받아서 Client 리스트 화면과 Client 가 접속하고 접속 해제하는 event 를 발생시키는 예제를 다루도록 하겠습니다.

3. CUSTOMER INFO

3.1 CUSTOMER INFO 정의

이번 챕터는 채팅에서의 nickname 등 그룹의 자체 정보가 필요한 경우에 대한 튜토리얼 입니다. 실제로 필요하지 않는 경우는 바로 다음 챕터를 보시면 됩니다.

CUSTOMER INFO 는 참여한 그룹에 Client 개인 정보입니다. DPMessage 에서 고객의 Business 를 적용 할 수 있는 유일한 수단입니다. 앞에서 간단한 채팅 예제 경우 완성도를 높이기 위해서 채팅에 관련된 nickname 이 필요할 수 있습니다. 이러한 데이터를 DPMessage Server 에 보내서 다른 Client 와 Sync 를 맞추는 작업이 필요합니다. 또한 Client 가 챗방에 접속하거나 접속이 끊겼을 경우 해당 이벤트에 대한 내용도 Client 들에게 전달해야 합니다. 아래 다이어그램은 CUSTOMER INFO 의 이벤트 발생에 대한 순서도를 보여줍니다.



CUSTOMER INFO 는 **그룹 단위의 정보**를 저장합니다. 같은 Client 에 속해 있는 모든 그룹은 그룹별로 서로 다른 CUSTOMER INFO 를 가질 수 있습니다.

3.2 CUSTOMER INFO 전달

앞에 1.3 에서 그룹을 선언할 경우 options, callback 을 받도록 되어 있었습니다. options 와 callback 은 CUSTOMER INFO 를 위해 세팅이 필요한 부분입니다.

```
var group = client.openGroup (groupname, options, callback);
```

options 에서 custevent 를 true 로 선언할 경우 (default 는 false) 특정 Client 의 그룹 오픈, CUSTOMER INFO 의 변경, Client 의 접속해제 등 3 가지 event 에 대한 정보를 그룹에서 받을 수 있게 됩니다. 채팅의 예에서 보면 채팅방에 참여하게 되면 해당 nickename 을 전체 Clients 에게 전송하고 해당 그룹에 참여하였음을 알려야합니다. 또한 중간에 nickname 을 변경했을 경우 해당 nickname 정보를 변경했음을 알려야 하고 챗방에서 나갈 경우 접속이 해제되었다는 정보도 알려야 합니다.
callback 을 좀 더 자세하게 설명하면 그룹에 접속한 후 접속한 Client 의 CUSTOMER INFO 가 DPMessage 에 저장됩니다. 그 다음 DPMessage 는 저장된 CUSTOMER INFO 를 모든 Client 에게 전송합니다. 각각의 Client 는 CUSTOMER INFO 를 받은 후에 callback 함수를 호출하게 됩니다. 해당 callback 이 완료 된 후 부터 group.getUserList() 함수를 통해서 전체 접속 Clients 의 CUSTOMER INFO 정보를 가져올 수 있게 됩니다. group.getUserList() 함수에 대한 자세한 정보는 3.4 에서 좀 더 자세하게 설명됩니다. 채팅의 예로 openGroup 을 다시 선언하면 다음과 같이 선언할 수 있습니다.

채팅 그룹의 선언)

```
var group = client.openGroup('chat-group', {
      'custevent' : true,
      'custinfo' : {'nickname' : '홍길동'},
   }, function(data) {
      // 챗방에 접속된 유저의 목록 표현
});
```

3.3 3 종류 이벤트 함수

3.3.1 group.onUserIn

특정 그룹에 Client 의 접속이 있을 경우 호출되는 함수 입니다.
기본선언은 다음과 같습니다.
group.onUserIn(event 발생시 전달받을 callback 함수);

```
group.onUserIn (callback);
```

- callback (Function) required
  특정 그룹에 Client 접속 시 해당 그룹에 CUSTOMER INFO 를 전달
  callback function parameter
  - userid (String) : 접속한 그룹의 개인 Qnique 한 ID 값 (시스템값)
  - custinfo (Object) : 접속한 그룹의 개인 CUSTOMER INFO

채팅의 예로 접속 Client 에 대한 이벤트에 대해서는 다음과 같이 선언할 수 있습니다.

채팅 Client 접속 시)

```
group.onUserIn(function(userid, custinfo) {
   // custinfo.nickname 으로 채팅 방에 입장했음을 알림
   // group.users()를 통해서 챗방에 접속된 유저의 목록 표현
});
```

3.3.2 group.onUserOut

Client 의 접속 해제의 경우 호출되는 함수 입니다.

기본선언은 다음과 같습니다.
group.onUserOut(event 발생시 전달받을 callback 함수);

```
group.onUserOut (callback);
```

- callback (Function) required
  client 의 접속 해제 시 그룹별 Client 의 CUSTOMER INFO 를 전달
  callback function parameter
  - userid (String) : 접속한 그룹의 개인 Qnique 한 ID 값 (시스템값)
  - custinfo (Object) : 접속 해제한 Client 의 CUSTOMER INFO

채팅의 예로 접속 해제 Client 에 대한 이벤트에 대해서는 다음과 같이 선언할 수 있습니다.

채팅 Client 접속 해제 시)

```
group.onUserOut(function(userid, custinfo) {
   // custinfo.nickname 으로 채팅 방에 퇴장했음을 알림
   // group.getUserList()를 통해서 챗방에 접속된 유저의 목록 표현
});
```

3.3.3 group.onUserUpdated

채팅방의 nickname 등의 그룹의 CUSTOMER INFO 가 변경되었을 경우 호출되는 함수 입니다.

기본선언은 다음과 같습니다.
group.onUserUpdated(event 발생시 전달받을 callback 함수);

```
group.onUserUpdated (callback);
```

- callback (Function) required
  그럽의 CUSTOMER INFO 변경 시 해당 Client 의 CUSTOMER INFO 를 전달
  callback function parameter
  - userid (String) : 접속한 그룹의 개인 Qnique 한 ID 값 (시스템값)
  - custinfo (Object) : Client 의 변경된 CUSTOMER INFO

채팅의 예로 nickname 을 변경한 Client 에 대한 이벤트에 대해서는 다음과 같이 선언할 수 있습니다.

채팅 nickname 변경 시)

```
group.onUserUpdated(function(userid, custinfo) {
   // custinfo.nickname 으로 nickname 이 변경 되었음을 알림
   // group.getUserList()를 통해서 챗방에 접속된 유저의 목록 표현
});
```

3.3.4 CUSTOMER INFO 의 userid

위 이벤트에서 callback 되는 CUSTOMER INFO 의 DATA 에는 UNIQUE 한 값인 userid 가 Callback 파라미터로 리턴됩니다.

userid 는 모든 Client 들에 대해서 UNIQUE 한 값임으로 업무 구현에 사용하셔도 됩니다. 한 Client 에 서로 다른 그룹에 대해서 userid 는 같은 값으로 생성됩니다. 대신 userid 는 reconnetion 되는 경우 새로운 값으로 변경됩니다.

3.4 변경된 정보 전달

3.4.1 변경된 CUSTOMER INFO 전달 함수 group.updateUser

채팅을 예로 들면 중간에 nickname 등의 CUSTOMER INFO 가 변경될 경우가 있습니다. 변경이 발생 될 경우 updateUser 를 통해서 DPMessage Server 에 해당 변경 사항을 알리고 다른 모든 Client 의 local CUSTOMER INFO 와 sync 를 맞추게 됩니다.
기본선언은 다음과 같습니다.
group.updateUser(변경된 CUSTOMER INFO 정보);

```
group.updateUser(custinfo);
```

- custinfo (Object) required
  변경된 CUSTOMER INFO

updateUser 가 호출된 경우 해당 함수를 호출한 client 를 포함 모든 client 그룹으로 onUserUpdated 이벤트가 발생하게 됩니다.
채팅의 예로 nickname 변경에 대해서는 다음과 같이 호출하면 됩니다.

채팅 nickname 변경 시 호출)

```
group.updateUser({
    'nickname' : '홍길동'
});
```

3.4.2 전체 CUSTOMER INFO 정보를 리턴하는 group.getUserList()

채팅을 예로 들면 Client 들의 nickname 을 표현해야 합니다. 전체 CUSTOMER INFO 는 DPMessage Server 와 각각의 Client local 사이에 언제나 Sync 를 하도록 되어 있습니다. group.getUserList()를 호출하는 경우 local 의 정보를 리턴하게 됩니다.
기본선언은 다음과 같습니다.

```
var userlist = group.getUserList();
```

전체 Client 의 CUSTINFO 를 반환은 다음과 같이 하면 됩니다.

전체 Client 의 CUSTINFO 를 반환하는 예)

```
var userlist = group.getUserList();
// 채팅 목록에 userlist 표현
```

CUSTOMER INFO 의 경우 DPMessage Server 와 각각의 Client 간의 정보 무결성을 완전히 보장하지는 못합니다. group.getallonUserInfo(callback)을 통해서 강제로 서버의 데이터와 Sync 를 맞출수 있습니다. 다만 네트웍의 부하를 유발 할 수 있는 함수인 관계로 차후에 deprecated 될 수 있습니다.

3.4.3 group.getUserList()의 DATA FORMAT

일반적인 CUSTOMER INFO 의 callback 함수는 userid 가 파라미터로 값을 전달 받습니다. 다만 group.getUserList()의 경우 목록을 표현해야 함으로 다음과 같은 DATA FORMAT 으로 리턴됩니다.

```
{
   '[userid 값 1]':  CUSTOMER INFO 값 1,
   '[userid 값 2]':  CUSTOMER INFO 값 2,
   ...
}
```

3.5 BUSINESS SERVER 의 연동

BUSINESS SERVER 는 4 장을 참고하시면 됩니다. 이해를 돕기 위해서 서버 API 라고 생각하시면 됩니다. 5 장에서 소개될 인증과정의 한 요소로 CUSTOMER INFO 를 서버에서 직접 정보를 전달 할 수 있습니다. 5 장에서 소개할 인증 과정에서 사용되는 함수는 다음과 같습니다. SERVER API 중 PHP 로 이루어진 샘플입니다.

```
$dpmessage-> genAuthKey(groupname, userid, custinfo);
```

- groupname (String) required
  임의의 그룹명
- userid (Object) required
  Client userid
- custinfo (Object) required
  접속한 Client 의 CUSTOMER INFO

특정 인증키를 발급 받을 때 userid 와 함께 custinfo 를 함께 전달 받습니다. 다만 인증과정은 그룹에 접속 시에 최초 과정에 속합니다. **그룹 선언 시에 CUSTOMER INFO 와 인증키를 받을 genAuthKey 에서의 CUSTOMER INFO 를 둘다 선언한 경우 Client 의 그룹 선언 시에 CUSTOMER INFO 가 적용됩니다.** 또한 userid 를 BUSINESS SERRVER 에서 직접 저장하고 있을 경우 중간에 CUSTOMER INFO 를 변경 할 수 있습니다.

```
$dpmessage->sendOnUserInfo(groupname, userid, custinfo);
```

- groupname (String) required
  임의의 그룹명
- userid (Object) required
  Client userid
- custinfo (Object) required
  변경된 CUSTOMER INFO

3.6 사용자 정보 채팅 예제

지금까지의 내용을 바탕으로 전에 구현했던 간단한 채팅에 사용자 정보를 추가해 보도록 하겠습니다.

자세한 html 은 custinfochat.html 을 참조하시면 됩니다.

"1.7 간단한 채팅 예제" 에서 바뀐 부분은 처음 페이지에 접속할 때 nickename, 나이, 성별을 입력 받는 화면과 오른쪽 하단에 change nickname 을 이용해서 접속 중에 해당 정보를 바꾸는 버튼, 오른쪽에 채팅방 접속 유저 리스트를 표현하는 부분이 추가되었습니다.

openGroup 부분은 다음과 같이 수정합니다.

```
client.openGroup('chat-group', {
    'custevent' : true,
    'custinfo' : custinfo,
    'clientevent' : true,
}, function(data) {
    // 사용자 목록 화면 그리기
});
```

다음의 3 가지 CUSTOMER INFO 함수를 추가합니다.

```
// 사용자 login 콜백
group.onUserIn(function(userid, custinfo) {
    // custinfo.nickname 님이 입장하셨습니다.
    // 사용자 목록 화면 그리기
});
// 사용자 logout 콜백
group.onUserOut(function(userid, custinfo) {
    // custinfo.nickname 님이 퇴장하셨습니다.
    // 사용자 목록 화면 그리기
});
// 다른 client 의 사용자 정보가 수정되었을 경우 해당 함수에 callback 된다.
group.onUserUpdated(function(userid, custinfo) {
    // custinfo.nickname 으로 대화명이 변경되었습니다.
    // 사용자 목록 화면 그리기
});
```

또한 nickname 등의 정보가 변경된 경우
group.updateUser(custinfo);
를 호출하고 사용자 목록화면을 그릴 경우 다음과 같이 하면 됩니다.

```
var _jsondata = group.getUserList();
for ( var userid in _jsondata) {
   // 유저 목록을 화면에 그린다.
}
```

해당 풀소스는 custinfochat.html 을 참고하시면 됩니다. 다음 챕터에서는 고객 서버에서 직접 각 Client 에 공지메시지를 Push 하는 과정에 대해서 다루도록 하겠습니다.

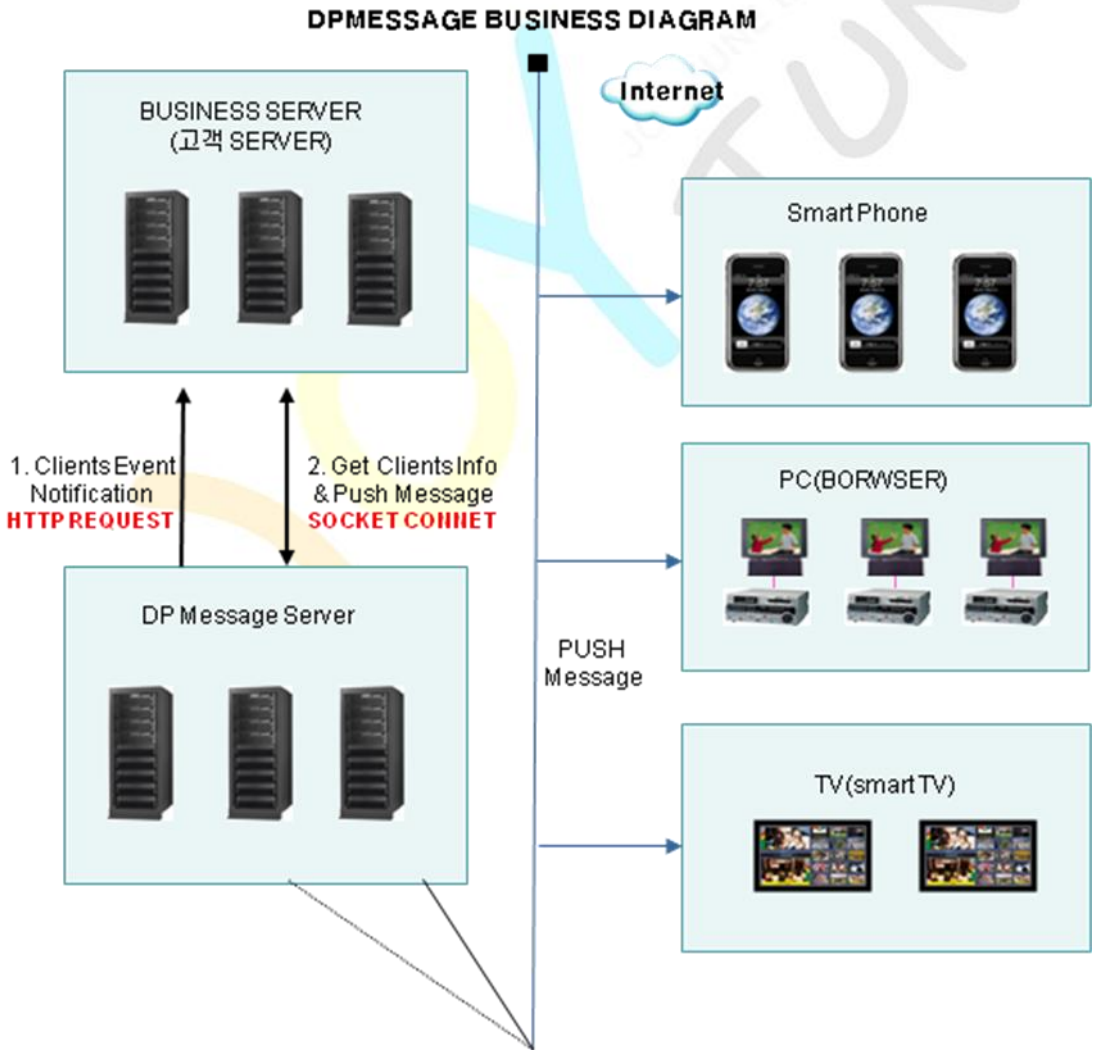
4. BUSINESS SERVER

4.1 BUSINESS SERVER 의 활용

BUSINESS SERVER 는 고객의 WEB APPLICATION SERVER 를 통해서 DPMESSAGE 의 여러가지 ACTION 이벤트를 수신하고 특정 ACTION 을 호출하며 인증등을 관리할 수 있는 서버입니다. BUSINESS SERVER 는 PRODUCTKEY 를 발급받은 고객이 직접 구현해야 하는 부분이며 현재 관리되고 있는 관리 서버에 API 를 제공하여 직접 개발이 가능하도록 되어 있습니다. API 는 JAVA, PHP 를 지원 중이며 ASP, nodejs 는 향후 지원할 예정입니다.

BUSINESS SERVER 는 다음 3 가지의 목적으로 활용될 수 있습니다.

- ✓ Client 들의 이벤트를 수신하고 Logging
- ✓ Client 의 정보를 가져오고 특정 Action 을 SEND
- ✓ PRODUCTKEY 와 그룹명을 통한 Client 그룹 인증



위 다이어그램은 BUSINESS SERVER 에서 구현해야 되는 작업을 전체 다이어그램을 통해서 간단히 표현한 내용입니다. PRODUCTKEY 의 Client 인증은 다음 챕터의 인증에서 상세히 다루도록 하겠습니다.

4.2 Clients Event Notification

DPMessage 에서 제공하는 Client 의 Action Event 는 다음의 3 가지입니다.

- ✓ Client 의 그룹 접속
- ✓ Client 의 접속 해제
- ✓ Client 의 Message Push

DPMessage 서비스 가입 후에 가입자 전용 페이의 PRODUCTKEY 설정 팝업에서 위의 3 가지 Action 이벤트에 대한 notification 을 받기 원하는 경우 해당 내용에 체크하고 Event Callback URL 을 등록하시면 HTTP REQUEST 로 BODY 에 JSON 형태의 DATA 로 전달되게 됩니다.

4.2.1 Client 의 그룹 접속

Client 의 새로운 접속이 있을 경우 각각의 Cient 에 group.onUserIn 이라는 callback 으로 해당 CUSTOMER INFO 의 정보가 들어오도록 되어 있습니다. 마찬가지로 새로운 접속이 있을 경우 BUSINESS SERVER 로 해당 내용을 전달하게 됩니다.

HTTP REQUET BODY (For User In)

- action (String)
  'onUserIn' 새로운 Client 의 접속
- groupname (String)
  접속한 그룹명
- userid (String)
  해당 Client 의 Unique 한 _userid (2.3.4 참조)
- onUserInfo (JSON 형태의 String)
  해당 Client 의 그룹에 대한 CUSTOMER INFO (2.1 참조)

Client 접속 시 전달 Sample)

```
{
'action' : 'userin',
'groupname' : 'chat-group',
'userid' : 'KYbfGCVxEbesAs7fAAAB',
'onUserInfo' : '{₩'nickname₩' : ₩'홍길동₩'}'
}
```

4.2.2 Client 의 접속 해제

Client 의 접속 해제가 있을 경우 각각의 Cient 에 group.onUserOut 이라는 callback 으로 해당 CUSTOMER INFO 의 정보가 들어오도록 되어 있습니다. 마찬가지로 접속이 해제가 되었을 경우 BUSINESS SERVER 로 해당 내용을 전달하게 됩니다. 다만 onUserOut 의 경우는 client 의 접속 해제가 이루어진 상태로 group 에 대한 정보가 아닌 userid 를 통한 전체 접속이 끊어졌음을 알립니다.

HTTP REQUET BODY (For User Out)

- action (String)
  'onUserIn' 새로운 Client 의 접속
- userid (String)
  해당 Client 의 Unique 한 _userid (2.3.4 참조)

Client 접속 해제 시 전달 Sample)

```
{
'action' : 'userout',
'userid' : 'KYbfGCVxEbesAs7fAAAB',
}
```

4.2.3 Client 의 Message Push

DPMesage 에서는 Client 에 Push 된 어떤 Message 정보도 저장하지 않습니다. 해당 내용에 대한 정보 저장도 고객의 선택사항입니다. 해당 메시지에 대해서 Push 가 일어나면 BUSINESS SERVER 로 해당 내용을 전달하게 됩니다.

HTTP REQUET BODY (For Push Message)

- action (String)
  'message' Push Message Action
- result (String)
  0 이면 성공 그 외의 값은 에러코드 참조
- userid (String)
  해당 Client 의 Unique 한 _userid (2.3.4 참조)
- onUserInfo (String)
  해당 Client 의 CUSTOMER INFO (2.1 참조)
- groupname (String)
  Message Push 된 Group Name
- actionname (String)
  Resceive 된 Action Name
- message (String)
  Client 에 Push 된 Message 내용

Message Push 전달 Sample)

```
{
'action' : 'message',
'result' : 0,
'userid' : 'KYbfGCVxEbesAs7fAAAB',
'onUserInfo' : '{₩'nickname₩' : ₩'홍길동₩'}'
'groupname' : 'chat-group',
'groupname' : 'chat-action',
'message' : 'Hello World!',
}
```

4.3 BUSINESS SERVER API

BUSINESS SERVER 의 활용 목적 중에 두번째로 Client 의 정보를 가져오고 특정 Action Message Push 하는 기능으로 다음의 3 가지 기본 SERVER API 를 제공합니다.

- ✓ send
- ✓ getGroupList
- ✓ getUserList

SERVER API 의 경우 PHP 소스를 기본으로 설명드립니다.

4.3.1 DPServer 의 선언

PHP 소스의 경우 DPMessage 에서 제공한 dpmessage.php 소스를 서버에 세팅 후에
include "[경로]/dpmessage.php";
를 선언합니다.

그 후에 기본선언은 다음과 같습니다.
$dpmessage = new DPServer('발급받은 PRODUCT KEY');

```
$dpmessage = new DPServer(productkey);
```

- productkey (String) required
  발급받은 Product Key

PHP sample)

```
$dpmessage = new DPServer('TESTCHAT');
```

4.3.2 send

1.5 챕터에서의 Client send 와 마찬가지로 Send Action 은 소속되어 있는 Group 에 해당 메시지를 보내는 행위입니다. 다만 SERVER API 의 경우 특정 그룹으로 JOIN 하는 형태가 아니라 DPMessage 의 instance 에서 직접 그룹명과 액션명을 지정해서 메시지를 PUSH 하게 됩니다.

기본선언은 다음과 같습니다.
$dpmessage->send('그룹명', '액션명', '데이터');

```
$dpmessage->send (groupname, actionname, pushdata);
```

- groupname (String) required
  임의의 그룹명
- actionname (String) required
  임의의 액션명
- pushdata (String) requried
  전달할 데이터의 내용

sample)

```
$dpmessage->send('chat-group', 'chat-action', 'Hello World!');
```

4.3.3 getGroupList

getGroupList 함수는 발급받은 PRODUCTKEY 로 현재 OPEN 되어 있는 전체 그룹리스트를 반환합니다. 그룹리스트는 그룹명을 키로 CUSTOMER INFO 사용 유무, CLIENT SEND 함수 사용유무를 리스트로 리턴하게 됩니다. PHP 의 경우 array 형태로 리턴합니다. 각각의 서버 언어마다 리턴하는 format 이 상이 할 수 있으니 SERVER API 를 참조하시기 바랍니다.

기본선언은 다음과 같습니다.
$dpmessage->getGroupList();

$dpmessage->getGroupList();

- return data (Array)
  그룹명 array
  ['그룹명'] => {['custevent']:CUSTOMERINFO 사용유무, ['sendevent']:CLIENT SEND 사용 유무 }

sample)
```
$grouplist = $dpmessage->getGroupList();
var_dump($grouplist);
// array 형태의 grouplist
```

4.3.4 getUserList

getUserList 함수는 CUSTOMER INFO 를 사용하는 경우에만 특정 그룹에 속해있는 전체 Client 리스트의 CUSTOMER INFO 를 반환합니다. CUSTOMER INFO 를 사용하지 않는 경우는 비어있는 리스트를 반환합니다. Client 가 세팅한 CUSTOMER INFO 와 함께 UNIQUE 한 _userid 가 같이 반환됩니다. 또한 CUSTONER INFO 를 세팅하지 않은 경우 _userid 만 리턴하게 됩니다. PHP 의 경우 array 형태로 리턴합니다. 각각의 서버 언어마다 리턴하는 format 이 상이 할 수 있으니 SERVER API 를 참조하시기 바랍니다.

기본선언은 다음과 같습니다.
$dpmessage->getUserList('그룹명');

$dpmessage->getUserList(groupname);

- groupname (String) required
  임의의 그룹명
- return data (Array)
  Client 정보 array $userlist['test-chat'] -> 각 Client 의 CUSTOMER INFO 배열리턴

sample)
```
$userlist = $dpmessage->getUserList();
```

```
var_dump($userlist);
// array 형태의 userlist
```

4.3.5 Debug For SERVER API

PHP 에 대한 Debug 를 가이드합니다. 각각의 서버 언어마다 Debug 하는 방법이 상이할 수 있으니 SERVER API 를 참조하시기 바랍니다. DPMessage 의 SERVER API 는 함수하나에 한번 이상의 socket connection 을 가지게 됩니다. 따라서 각 함수의 실행이 끝난 후에 성공, 실패에 대한 return 은 **실행문 바로 다음에 삽입**해야 합니다. getResult 함수를 이용해서 Debug 를 할 수 있습니다. getResult 함수는 'ERRORCODE', 'ERRORDESC' 를 포함하는 Array 형태의 data 를 리턴하며 ERRORCODE 가 0 인 경우는 성공입니다. 그 외의 값에 대한 부분은 에러코드를 참조하시면 됩니다.

기본선언은 다음과 같습니다.
$dpmessage->getResult();

$dpmessage->getResult();

- return data (Array)
  실행결과 array 'ERRORCODE', 'ERRORDESC'

sample)

```
$dpmessage = new DPServer("TESTCHAT");
var_dump($dpmessage->getResult());  // new DPServer 선언에 대한 Debug
$dpmessage->send('chat-group','chat-action-notice', "Hello World From SERVER");
var_dump($dpmessage->getResult());  // send 실행에 대한 Debug
$groupList = $dpmessage->getGroupList());
var_dump($dpmessage->getResult());  // getGroupList 실행에 대한 Debug
```

4.4 공지 메시지를 포함한 채팅

지금까지의 내용을 바탕으로 샘플로 구현된 채팅에 공지를 추가해보도록 하겠습니다.

기존의 채팅 html 안에 공지내용을 화면에 보여줄 수 있도록 특정 action 을 onReceive 합니다.

```
group.onReceive('chat-action-notice', function(data, userid, custinfo) {
   writeChat('<font color=blue><b>[공지] : ' + data + '</b></font>');
});
```

PHP 서버의 특정 php 파일을 생성 후에 다음의 소스를 추가합니다.

```
include "dpmessage.php";
$dpmessage = new DPServer("TESTCHAT");
$dpmessage->send('chat-group','chat-action-notice',"알려드립니다.  테스트  공지입니다.");
//echo "=== groupList ===<br>";
//var_dump($dpmessage->getGroupList());  // 그룹리스트
//echo "<br><br>=== userList ===<br>";
```

```
//var_dump($dpmessage->getUserList('chat-group'));  // 유저리스트
```

실제로는 샘플 소스 상에 3 번째 줄까지만 입력하면 공지가 채팅 Client 들에게 전달되며 그 뒤의 내용은 getGruopList 와 getUserList 에 대한 샘플입니다. 해당 php 파일에 간단한 input box 를 추가해서 공지를 입력할 수 있도록 변경하는 작업도 가능합니다. 지금까지 BUSINESS SERVER 를 이용한 채팅공지를 만들어 봤고 다음 챕터에서는 Client 를 인증하는 과정에 대한 내용을 다루도록 하겠습니다.

5. 보안/인증

5.1 보안

DPMessage 에서 제공하는 보안은 SSL 보안입니다. 챕터 1.2 에서 Client 의 선언 시에 옵션에 security 항목이 있었습니다.

```
var client = new DPClient(productkey, options);
```

- productkey (String) required
  발급받은 Product Key
- options (Object) optional
  DPMessage 옵션정보
  - security (Boolean) : SSL 보안을 사용할지의 유무
  - resultcallback (Function) : debug 용 callback 함수

SSL 보안을 사용하는 방법은 간단하며 security option 을 true 로 해주시면 됩니다.
{'security' : true}
암호화 구간은 Client 에서 DPMessage 서버 사이의 양 구간이며 Bisiness Server 의 경우 암호화를 차후에 지원할 예정입니다. 또한 Bisiness Server 의 경우 특정 IP 로만 접근 가능하도록 하는 다른 형태의 보안 역시 지원될 예정입니다.
현재 Close 버전에서는 누구나 보안 사용이 간단합니다. 다만 차후에 정식 서비스에는 암호화에 대한 지원은 유료 옵션으로 사용될 예정입니다.

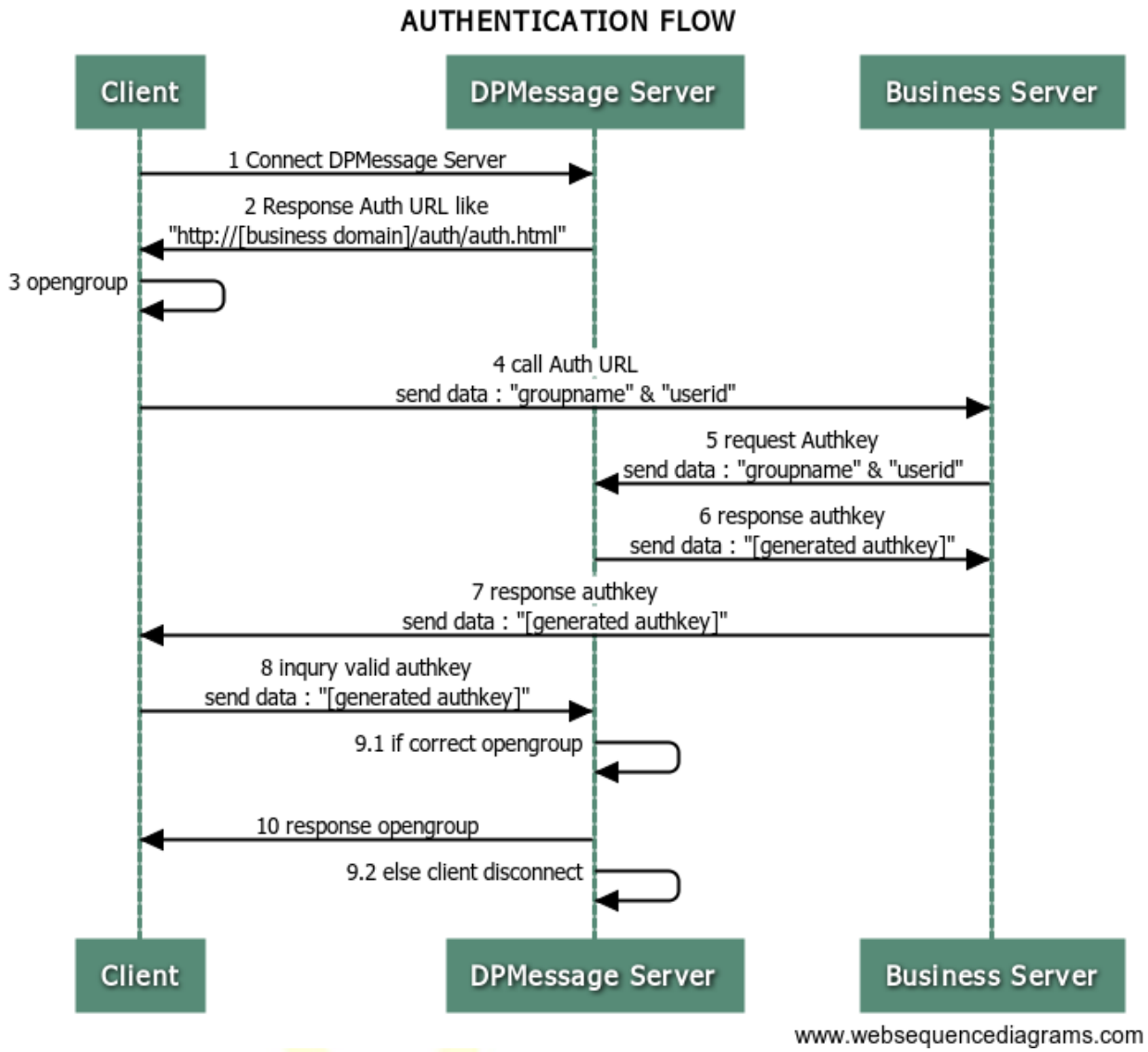
5.2 인증

5.2.1 인증을 위한 준비

PRODUCTKEY 는 일반키와 인증키의 2 가지 타입이 있습니다. 기본적으로 제공되는 키는 일반키이며 일반키는 인증과정을 거치지 않습니다. 가입자 전용 페이의 PRODUCTKEY 설정 팝업에서 인증키로 변경할 수 있습니다. 인증키의 경우 해당 그룹에 접속 할 경우 (openGroup) 인증과정을 무조건 거치도록 되어 있습니다. 인증을 위해서는 BUSINESS SERVER 의 인증 URL 을 입력해야 합니다. DPMessage 에서 로그인 하시면 해당 인증키를 발급받으실 수 있으며 발급받은 인증키에 대한 인증 URL 을 설정 화면에서 입력하시면 됩니다.

5.2.2 DPMessage 에서의 인증

DPMessage 에서의 인증은 일반적인 user 에 대한 Business 인증이 아닙니다. 해당 Client 에 ProductKey, groupname, userid 를 기반으로 최초 접속에 대한 인증키를 발급함으로써 다른 패킷 변조등을 통해 해당 Client 인듯 속이고 들어올 수 없도록 인증하는 형태입니다. 인증은 그룹 단위로 이루어지며 인증키를 1 차로 BISINESS SERVER 와 DPMessage 서버의 통신을 통해서 발급받으며 해당 인증키를 Client 로 전달하여 DPMessage 서버와의 인증을 하는 형태입니다.



5.2.3 인증을 위한 함수

openGroup 의 파라미터로 다음과 같이 선언하도록 되어 있습니다.

```
var group = client.openGroup (groupname, options, callback);
```

- options (Object) optional
  DPMessage 옵션정보
  - custevent (Boolean) : 그룹에서 CUSTMER INFO 이벤트를 callback 받는 유무
  - custinfo (Object) : 연결된 Client 의 CUSTOMER 정보
  - authparam (String) : 인증 URL 전달 파라미터
  - authcallback (Function) : 인증 후 실행될 callback 함수

options 에서 authparam 과 authcallback 두가지를 제공하는데 이것은 5.2.4 에서 설명할 BUISNESS 인증을 위해서 필요한 내용입니다. Client 에서는 다른 부분에 대한 세팅이 필요없습니다.

서버 API 의 경우 등록된 인증 URL 의 서버 프로그램에서 다음의 샘플처럼 함수를 CALL 하시면 됩니다. PHP 소스를 샘플로 하도록 하겠습니다.

PHP sample)

```
include "dpmessage.php";
$dpmessage = new DPServer("TESTCHAT");
echo $dpmessage->genAuthKey($_REQUEST['groupname'], $_REQUEST['userid'], array());
```

기본선언은 다음과 같습니다.
$dpmessage->genAuthKey('그룹명', 'userid', 'CUSTOMER INFO 정보');

```
$dpmessage->genAuthKey (groupname, userid, custinfo);
```

- groupname (String) required
  전달받은 그룹명
- userid (String) required
  전달받은 userid
- custinfo (String) requried
  접속하려는 user 의 CUSTOMER 정보

위의 소스 내용을 그대로 카피하셔서 인증 URL 서버 페이지에 넣으시면 DPMessage 인증과정은 완료됩니다. CUSTOMER INFO 를 인증 시에 서버에서 바로 넘길 수 있습니다. BUSINESS SERVER 가 있는 경우 CUSTOMER INFO 를 Client 가 아닌 SERVER 에서 바로 DPMessage 에 전달이 가능합니다. 다만 Client 와 BUINSESS SERVER 둘다 CUSTOMER INFO 가 선언되어 있는 경우 Client 의 데이터를 우선으로 합니다.

5.2.4 BUSINESS 인증 방법

실제로 인증은 사이트에서 로그인 등의 인증 후에 받은 세션, 쿠키 등의 서버 스토리지에 저장된 값으로 이루어집니다. DPMessage 에서는 이러한 BUSINESS 인증을 자체적으로 지원하지 않습니다. 고객이 직접 인증 URL 에서 세션, 쿠키 혹은 도메인을 기반으로 직접 인증을 해야 합니다. 인증 URL 에 전달되는 내용을 토대로 접속 가능 여부를 인증 URL 서버 프로그램에서 직접 작성하셔야 합니다. 2.4.3 에서의 예제를 바탕으로 간단히 다음과 같이 하시면 됩니다.

PHP sample)

```
include "dpmessage.php";
$dpmessage = new DPServer("TESTCHAT");

// TODO 여기에 인증관련 비지니스 로직 추가

if([인증여부]) { // 만약 인증된 유저면
    echo      $dpmessage->genAuthKey($_REQUEST['groupname'],      $_REQUEST['userid'],
array());
} else {  // 인증이 안된 유저면
    echo "";
}
```

BUSINESS 인증을 위해서 Client 에 openGroup 함수에서 다음의 두가지 option 을 제공합니다.
authparam 은 Client 에서 BUSINESS 인증을 위해 넘겨야 할 데이터가 존재 할 경우 사용됩니다. String 형태로 다음과 같이 정의 하시면 됩니다. {'authparam' : "param1=param1&param2=param2"} 데이터의 전달은 POST 형태로 전달됩니다.
다른 옵션인 authcallback 의 경우 인증 후에 처리해야 할 일이 있을 경우 사용됩니다. Function 형태로 다음과 같이 정의하시면 됩니다. {'authcallback' : [callback function]}
간단한 예로 다음과 같이 사용하시면 됩니다.

sample)

```
var authcallback = function(resultcode, resultdesc) {
      if(resultcode == 0) {
         alert("인증이 완료되었습니다.");
      } else {
         alert("인증이 실패했습니다. : " + resultdesc);
      }
}
group = client.openGroup('chat-group', {
      'authparam' : "param1=param1&param2=param2",
      'authcallback' : authcallback
   });
```

6. PRODUCTKEY 설정

6.1 PRODUCTKEY 설정방법

www.dpush.co.kr 의 사이트에서 로그인을 합니다. 로그인 후에 첫화면 "PRODUCTKEY" 메뉴 화면을 보면 등록한 PRODUCTKEY 의 리스트를 보여줍니다. 해당 PRODUCTKEY 화면에서 설정을 클릭합니다.

PRODCUTKEY 에 대한 설정화면이 다음과 같이 나타납니다.

✓ PRODUCT KEY 명
키에 대한 구분을 위한 이름으로 임의의 키명으로 세팅하시면 됩니다.

✓ 종류
기본은 일반키이며 인증이 필요한 경우 인증키를 선택하시면 됩니다. 인증의 경우 5.2 절을 참고하시면 됩니다.

✓ 인증 Uri
인증키의 경우 활성화 되며 인증을 수행할 Business Server 의 Uri 를 지정합니다.

✓ Customer Info 사용
기본은 사용안하도록 되어 있으며 사용을 원하시는 경우 체크하시면 됩니다. **체크가 안되어 있는 경우 openGroup 시에 옵션에 custevent 를 true 로 설정하셔도 적용되지 않습니다.** Customer Info 를 사용하는 경우 Client 의 수가 많으면 메시지 카운트를 많이 증가시키게 됩니다. Customer Info 는 3 장을 참고하시면 됩니다.

✓ 클라이언트 메시지
Client 에서 send 메시지를 보낼수 있도록 설정합니다.
기본은 사용안하도록 되어 있으며 사용을 원하시는 경우 체크하시면 됩니다. **체크가 안되어 있는 경우 openGroup 시에 옵션에 sendevent 를 true 로 설정하셔도 적용되지 않습니다.**

✓ Event Callback 사용
Client 의 그룹접속, 해제, 메시지전송 등의 이벤트가 일어날 경우 등록한 Business Server Uri 로 해당 이벤트 내용을 전달해 줍니다. Event Callback 은 4.2 절을 참고하시면 됩니다.

✓ Event Callback Uri
Event Callback 을 하나라도 체크했을 경우 활성화 되며 이벤트를 수신할 Business Server 의 Uri 를 지정합니다.

7. ERROR CODE

Error Code	Description
9999	알수없는에러
1000001	유저이벤트를받지않도록설정되어있음
3001001	등록되지않은상품키
3001002	사용기간만료
3001003	접속유저수초과
3001004	인증 URL 미등록
3001005	보안사용불가
3003001	그룹생성실패
3003002	인증 URL 접속실패
3003003	인증페이지에러
3003004	인증되지않은유저
3003005	최초생성된그룹과다른유저이벤트(custevent 정보가다름)
3003006	최초생성된그룹과다른 client 이벤트(sendevent 정보가다름)
3004001	메세지전송량초과(하루단위)
3004002	client 에서 send 를하지못하도록설정(sendevent 가 false)
3004003	메세지용량(byte)초과
3010001	인증되지않은접속