1 群组管理 REST API

群组管理 REST API 主要分为以下几类：

1. 获取 APP 中的所有群组；
2. 获取群组相关信息，包括如下协议：
   a) 获取群组详细资料；
   b) 获取群成员详细资料；
   c) 获取某一用户所加入的群组；
3. 群组管理操作，包括如下协议：
   a) 创建群组；
   b) 解散群组；
   c) 修改群组基础资料；
   d) 修改群成员资料；
   e) 增加群组成员；
   f) 删除群组成员；
   g) 禁言/取消禁言；
4. 消息下发操作，包括如下协议：
   a) 向群组发送普通消息；
   b) 向群组发送系统通知。

1.1 获取 APP 中的所有群组

ServiceName	group_open_http_svc
Command	get_appid_group_list
协议类型	公开协议
接口说明	1. 该接口返回 APP 中的所有群组； 2. 如果 APP 中的总群数量超过 10000 个，最多只会返回 10000 个（如果需要获取完整必须使用高级接口），此时返回包中的 TotalCount 字段表示的是真正的群组总数。也就是说，如果 APP 中一共有 50000 个群，那么在调用该接口时，响应包体中的 TotalCount 将为 50000，但 GroupIdList 数组中仅会有 10000 个群组 ID； 3. 如果需要分页拉取，或者依照群组形态过滤，请使用高级接口。
请求包体	{} // 空 Json 对象，注意不能不填
应答包体	{ "ActionStatus": "OK", "ErrorInfo": "", "ErrorCode": 0, "TotalCount": 2, // APPID 下的群组总数量 "GroupIdList": [{ "GroupId": "@TGS#2J4SZEAEL" }, {

	"GroupId": "@TGS#2C5SZEAEF" }] }

1.2 获取 APP 中的所有群组（高级接口）

ServiceName	group_open_http_svc
Command	get_appid_group_list
协议类型	公开协议
接口说明	1. 该协议是对简单接口的扩充，支持分页拉取，支持依照群组形态过滤； 2. Limit 和 Offset 两个值用于控制分页拉取： a) Limit 限制回包中 GroupIdList 数组中群组的个数，不得超过 10000； b) Offset 控制从整个群组列表中的第多少个开始读取从 0 开始。对于分页请求（页码数字从 1 开始），每一页的 Offset 值应当为：（页码数 – 1）× 每页展示的群组数量； c) 例如：假设需要分页拉取，每页展示 20 个，则第一页的请求参数应当为：{“Limit” : 20, “Offset” : 0}，第二页的请求参数应当为{“Limit” : 20, “Offset” : 20}，依此类推； d) Limit 或者 Offset 的取值不会对应答包体中的 TotalCount 造成影响； 3. 如果仅需要返回特定群组形态的群主，可以通过 GroupType 进行过滤，但此时返回的 TotalCount 的含义就变成了 APP 中该群组形态的群组总数。例如：假设 APP 中总共 50000 个群组，其中有 20000 个为公开群组，如果将请求包体中的 GroupType 设置为 Public，那么不论 Limit 和 Offset 怎样设置，应答包体中的 TotalCount 都为 20000，且 GroupIdList 中的群组全部为公开群组。

请求包体	{
	"Limit": 1000, // 最多获取多少个群，不得超过 10000（选填）
	"Offset": 2000, // 从第多少个开始获取（选填）
	"GroupType" : "Public" // 仅获取指定群组形态的群组（选填）
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0,
	"TotalCount": 2, // APPID 下的群组总数，或者指定群组形态的群组总数
	"GroupIdList": [
	{
	"GroupId": "@TGS#2J4SZEAEL"
	},
	{
	"GroupId": "@TGS#2C5SZEAEF"
	}
	]
	}

1.3 创建群组

ServiceName	group_open_http_svc
Command	create_group
协议类型	公开协议
接口说明	1. 支持创建私有群、公开群、聊天室三种群组。
请求包体	{ "Owner_Account": "leckie", // 群主的 UserId（选填） "Type": "Public", // 群组类型：Private/Public/ChatRoom（必填） "Name": "TestGroup", // 群名称（必填） "Introduction": "This is group Introduction", // 群简介（选填） "Notification": "This is group Notification", // 群公告（选填） "FaceUrl": "http://this.is.face.url", // 群头像 URL（选填） "MaxMemberCount": 500,// 最大群成员数量（选填） "ApplyJoinOption": "FreeAccess", // 申请加群处理方式（选填） "AppDefinedData": [// 群组维度的自定义字段（选填） { "Key": "GroupTestData1", // APP 自定义的字段 Key "Value": "xxxxx" // 自定义字段的值 }, { "Key": " GroupTestData2", "Value": "abc\u0000\u0001" // 自定义字段支持二进制数据 }], "MemberList": [// 初始群成员列表，最多 500 个（选填） { "Member_Account": "bob", // 成员（必填） "Role": "Admin" // 赋予该成员的身份，目前备选项只有 Admin（选填） }, { "Member_Account": "peter" }] }
应答包体	{ "ActionStatus": "OK", "ErrorInfo": "", "ErrorCode": 0, "GroupId": "@TGS#2J4SZEAEL" // 创建成功之后的群 ID，由 IM 云后台分配 }

1.4 获取群组详细资料

ServiceName	group_open_http_svc
Command	get_group_info
协议类型	公开协议
接口说明	1. 该接口支持批量获取一批群的全部资料； 2. 出于省流量等目的，如果只需返回群组的部分基础信息或者成员信息，请使用高级接口； 3. 如果需要分页拉取群成员，请使用获取群组成员详细资料接口； 4. 如果一次请求的群组数量过多，或者群组本身成员数量较大，极有可能导致响应包过大，超出系统的响应包长度限制（1MB），此时返回码为 10018；为避免出现这种情况，建议尽可能使用使用高级接口精确指定需要返回的数据，或者减少需要拉取的群组数量。

请求包体	{
	"GroupIdList": [// 群组列表（必填）
	"@TGS#2J4SZEAEL",
	"@TGS#2C5SZEAEF"
	]
	}

应答包体	{ "ActionStatus": "OK", "ErrorInfo": "", //这里的 ErrorInfo 无意义，需要判断每个群组的 ErrorInfo "ErrorCode": 0, // 这里的 ErrorCode 无意义，需要判断每个群组的 ErrorCode "GroupInfo": [// 返回结果为群组信息数组，为简单起见这里仅列出一个群 { "GroupId": "@TGS#2J4SZEAEL", "ErrorCode": 0, // 针对该群组的返回结果 "ErrorInfo":"" , // 针对该群组的返回结果 "Type": "Public", // 群组类型 "Name": "MyFirstGroup", // 群组名称 "Introduction": "TestGroup", // 群组简介 "Notification": "TestGroup", // 群组通知 "FaceUrl": "http://this.is.face.url", // 群组头像 URL "Owner_Account": "leckie", // 群主 ID "CreateTime": 1426976500, // 群组创建时间(UTC 时间) "LastInfoTime": 1426976500, // 最后群资料变更时间(UTC 时间) "LastMsgTime": 1426976600, // 群内最后一条消息的时间(UTC 时间) "NextMsgSeq": 1234, // "MemberNum": 2, // 当前群成员数量 "MemberNumMax": 50, // 最大群成员数量 "ApplyJoinOption": "FreeAccess", // 申请加群处理方式 "AppDefinedData": [// 群组维度的自定义字段 { "Key": "GroupTestData1", // 自定义字段的 key "Value": "xxxx" // 自定义字段的值

	}, {
	"Key": "GroupTestData2",
	"Value": "abc\u0000\u0001" // 自定义字段支持二进制数据
	}
	],
	"MemberList": [// 群成员列表
	{
	"Member_Account": "leckie", // 成员 ID
	"Role": "Owner", // 群内角色
	"JoinTime": 1425976500, // 入群时间(UTC 时间)
	"MsgSeq": 1233,
	"MsgFlag": "AcceptAndNotify", // 消息屏蔽选项
	"LastSendMsgTime": 1425976500, // 最后发言时间(UTC 时间)
	"ShutUpUntil": 1431069882 // 禁言截至时间(UTC 时间)
	}, {
	"Member_Account": "peter",
	"Role": "Member",
	"JoinTime": 1425976500, // 入群时间
	"MsgSeq": 1233,
	"MsgFlag": "AcceptAndNotify",
	"LastSendMsgTime": 1425976500, // 最后一次发消息的时间
	"ShutUpUntil": 0 // 0 表示未被禁言，否则为禁言的截止时间
	}
	]
	}
	]
	}

1.5 获取群组详细资料（高级接口）

ServiceName	group_open_http_svc
Command	get_group_info
协议类型	公开协议
接口说明	1. 高级接口支持仅获取群组中的指定字段，通过过滤器 ResponseFilter 来控制； 2. ResponseFilter 包含三个子过滤器，分别针对群基础资料、群成员、群维度的自定义字段，每个请求可以指定一个或者多个子过滤器； 3. 群基础资料、群成员子过滤器中，每个字段的含义详见群组系统综述； 4. 举例： a) 如果只需要获取群组名称，则 ResponseFilter 下仅有 GroupBaseInfoFilter 这一个子过滤器，该子过滤器的值为字符串数组，只有一个元素“Name”； b) 如果需要获取群组名称，以及群组维度的某个自定义字段，则 ResponseFilter 下应该有 GroupBaseInfoFilter 和 AppDefinedDataFilter_Group 这两个过滤器。
请求包体	{

	"GroupIdList": [// 群组列表，必填
	"@TGS#2J4SZEAEL",
	"@TGS#2C5SZEAEF"
	],
	"ResponseFilter": { //返回
	"GroupBaseInfoFilter": [// 如果基础信息字段，请在此数组中添加
	"Type",
	"Name",
	"Introduction",
	"Notification",
	"FaceUrl",
	"CreateTime",
	"Owner_Account",
	"LastInfoTime",
	"LastMsgTime",
	"NextMsgSeq",
	"MemberNum",
	"MaxMemberNum",
	"ApplyJoinOption"
	],
	"MemberInfoFilter": [// 如果需要成员信息，请添加此数组
	"Account", // 成员 ID
	"Role",
	"JoinTime",
	"MsgSeq",
	"MsgFlag",
	"LastSendMsgTime",
	"ShutUpUntil"
	],
	"AppDefinedDataFilter_Group": [// 群组维度的自定义字段过滤
	"GroupTestData1",
	"GroupTestData2"
	]
	}
	}

应答包体	// 应答包体与简单接口的格式相同，只是可能会少一些信息

1.6 获取群组成员详细资料

ServiceName	group_open_http_svc
Command	get_group_member_info
协议类型	公开协议
接口说明	1. 该接口的目的在于分页拉取单个群的成员列表，同样支持调用者指定需要返回哪些信息； 2. 此处的分页方法（Limit 与 Offset）与获取 APP 中的所有群组相同。

请求包体	{
	"GroupId": "@TGS#2J4SZEAEL", // 群组 ID（必填）
	"MemberInfoFilter": [// 需要获取哪些信息，如果没有该字段则为群成员全部资料
	"Account", // 成员 ID
	"Role",
	"JoinTime",
	"MsgSeq",
	"MsgFlag",
	"LastSendMsgTime",
	"ShutUpUntil"
	],
	"Limit": 100, // 最多获取多少个成员
	"Offset": 200 // 从第多少个开始获取
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0,
	"MemberNum": 2, // 本群组的群成员总数
	"MemberList": [// 群成员列表
	{
	"Member_Account": "bob",
	"Role": "Owner",
	"JoinTime": 1425976500, // 入群时间
	"MsgSeq": 1233,
	"MsgFlag": "AcceptAndNotify",
	"LastSendMsgTime": 1425976500, // 最后一次发消息的时间
	"ShutUpUntil": 1431069882 // 禁言截至时间(秒数)
	},
	{
	"Member_Account": " peter ",
	"Role": " Member ",
	"JoinTime": 1425976500,
	"MsgSeq": 1233,
	"MsgFlag": "AcceptAndNotify",
	"LastSendMsgTime": 1425976500,
	"ShutUpUntil": 0// 0 表示未被禁言，否则为禁言的截止时间
	}
	]
	}

1.7 修改群组基础资料

ServiceName	group_open_http_svc
Command	modify_group_base_info

协议类型	公开协议
接口说明	

请求包体	{
	"GroupId": "@TGS#2J4SZEAEL", // 要修改哪个群的基础资料（必填）
	"Name": "NewName", // 群名称（选填）
	"Introduction": "NewIntroduction", // 群简介（选填）
	"Notification": "NewNotification", // 群公告（选填）
	"FaceUrl": "http://this.is.new.face.url", // 群头像（选填）
	"MaxMemberNum": 500, // 最大群成员数量（选填）
	"ApplyJoinOption": "NeedPermission", // 申请加群方式（选填）
	"AppDefinedData": [// 自定义字段（选填）
	{
	"Key": "GroupTestData1", // 需要修改的自定义字段 key
	"Value": "NewData" // 自定义字段的新值
	},
	{
	"Key": "GroupTestData2",
	"Value": "" // 设置为空表示删除该项自定义字段
	}
	]
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0
	}

1.8 增加群组成员

ServiceName	group_open_http_svc
Command	add_group_member
协议类型	公开协议
接口说明	

请求包体	{
	"GroupId": "@TGS#2J4SZEAEL", // 要操作的群组（必填）
	"MemberList": [// 一次最多添加 500 个成员
	{
	"Member_Account": "tommy" // 要添加的群成员 ID（必填）
	},
	{
	"Member_Account": "jared"
	}
	]
	}

应答包体	{

	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0,
	"MemberList": [
	{
	"Member_Account": "tommy",
	"Result": 1 // 加人结果：0 为失败；1 为成功；2 为已经是群成员
	},
	{
	"Member_Account": "jared",
	"Result": 1
	}
	]
	}

1.9 删除群组成员

ServiceName	group_open_http_svc
Command	delete_group_member
协议类型	公开协议
接口说明	1. 只要操作者的权限足够，即使被删除的用户原本就不在群组中，该接口依然返回成功

请求包体	{
	"GroupId": "@TGS#2J4SZEAEL", //要操作的群组（必填）
	"Silence": 1, // 是否静默加人（选填，目前不支持）
	"MemberToDel_Account": [// 要删除的群成员列表，最多 500 个
	"tommy",
	"jared"
	]
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0
	}

1.10 修改群成员资料

ServiceName	group_open_http_svc
Command	modify_group_member_info
协议类型	公开协议
接口说明	
请求包体	{ "GroupId": "@TGS#2CLUZEAEJ", //要操作的群组（必填） "Member_Account": "bob", // 要操作的群成员（必填）

	"Role": "Admin", // Admin 或者 Member，分别为设置/取消管理员 "MsgFlag": "AcceptAndNotify", // 消息屏蔽类型 "ShutUpTime" : 60 // 禁言时间 }

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0
	}

1.11 解散群组

ServiceName	group_open_http_svc
Command	destroy_group
协议类型	公开协议
接口说明	群组解散之后将无法恢复，请谨慎调用该接口。

请求包体	{
	"GroupId": "@TGS#2J4SZEAEL"
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0
	}

1.12 获取某一用户所加入的群组

ServiceName	group_open_http_svc
Command	get_joined_group_list
协议类型	公开协议
接口说明	1. 高级接口进一步支持： a) 获取群组简单资料； b) 获取该用户在群内的资料； c) 分页拉取； d) 依照群组形态过滤。

请求包体	{
	"Member_Account": "leckie"
	}

应答包体	{ "ActionStatus": "OK", "ErrorInfo": "", "ErrorCode": 0, "TotalCount": 2, "GroupIdList": [

	{
	"GroupId": "@TGS#2J4SZEAEL"
	},
	{
	"GroupId": "@TGS#2C5SZEAEF"
	}
	]
	}

1.13 获取某一用户所加入的群组（高级接口）

ServiceName	group_open_http_svc
Command	get_joined_group_list
协议类型	公开协议
接口说明	1. 该接口支持分页拉取（通过请求包中的 Limit 和 Offset 控制），分页方式与获取 APP 中的所有群组相同； 2. 如果请求包中指定了 ResponseFilter，表示需要进一步获取群组的基本资料，以及该用户在每个群组中的个人资料； 3. 如果请求包中指定了 GroupType 这个过滤条件，则表示仅获取特定群组形态的群组； 4. 返回包中的 TotalCount，表示的是符合条件的所有群组，与是否启用分页无关。例如，即使指定 Limit 为 10，GroupType 为 Public，但如果用户加入的公开群组为 100 个，则回包中 TotalCount 的值为 100，但 GroupIdList 数组中只会有 10 个群组的信息。

请求包体	{
	"Member_Account": "leckie",
	"Limit": 10, // 拉取多少个，不填标识拉取全部
	"Offset": 0, // 从第多少个开始拉取
	"GroupType": "Public", // 拉取哪种群组形态，不填为拉取所有
	"ResponseFilter": {
	"GroupBaseInfoFilter": [// 需要哪些基础信息字段，含义参见“基本字段介绍”
	"Type",
	"Name",
	"Introduction",
	"Notification",
	"FaceUrl",
	"CreateTime",
	"Owner_Account",
	"LastInfoTime",
	"LastMsgTime",
	"NextMsgSeq",
	"MemberNum",
	"MaxMemberNum",
	"ApplyJoinOption"
	],
	"SelfInfoFilter": [// 自身在群内的信息
	"Role", // 群内身份

	"JoinTime", // 入群时间 "MsgFlag", // 消息屏蔽类型 "UnreadMsgNum" // 未读消息数量（IM 云后台计算得出）] } }
应答包体	

1.14 查询一组用户在某个群组中的身份

ServiceName	group_open_http_svc
Command	get_role_in_group
协议类型	公开协议
接口说明	1. 该接口类的作用相当于轻量级身份判定接口，即无需拉取所有群组成员，便可直接获取一批用户在群内的身份。

请求包体	{
	"GroupId": "@TGS#2C5SZEAEF",
	"User_Account": [// 最多支持 500 个
	"leckie",
	"peter",
	"wesley"
	]
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0,
	"UserIdList": [// 结果
	{
	"Member_Account": "leckie",
	"Role": "Owner" // 身份：Owner/Admin/Member/NotMember
	},
	{
	"Member_Account": "peter",
	"Role": "Member"
	},
	{
	"Member_Account": "wesley",
	"Role": "NotMember"
	}
	]
	}

1.15 批量禁言/取消禁言

ServiceName	group_open_http_svc
Command	forbid_send_msg
协议类型	公开协议
接口说明	1. 所谓禁言，即禁止某些用户在一段时间内发言。

请求包体	{
	"GroupId": "@TGS#2C5SZEAEF",
	"Members_Account": [// 最多支持 500 个
	"peter",
	"leckie"
	],
	"ShutUpTime": 60 // 禁言多少秒，为 0 表示取消禁言
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0
	}

1.16 在某一群组中发送普通消息

ServiceName	group_open_http_svc
Command	send_group_msg
协议类型	公开协议
接口说明	1. “普通消息”与“系统通知”的区别参见附录 1。

请求包体	{
	"GroupId": "@TGS#2C5SZEAEF",
	"MsgBody": [// 消息体， 由一个 element 数组组成，详见 TIMMessage 消息对象
	{
	"MsgType": "TIMTextElem", // 文本
	"MsgContent": {
	"Text": "red packet"
	}
	},
	{
	"MsgType": "TIMFaceElem", // 表情
	"MsgContent": {
	"Index": 6,
	"Data": "abc\u0000\u0001"
	}
	}
	]
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0
	}

1.17 在某一群组中发送系统通知

ServiceName	group_open_http_svc
Command	send_group_system_notification
协议类型	公开协议
接口说明	1. “普通消息”与“系统通知”的区别参见附录 1。

请求包体	{
	"GroupId": "@TGS#2C5SZEAEF",
	"ToMembers_Account": [// 接收者群成员列表，默认为空表示全员下发
	"peter",
	"leckie"
	],
	"Content": "Some info is Forbidden" // 系统通知内容，支持二进制数据
	}

应答包体	{
	"ActionStatus": "OK",
	"ErrorInfo": "",
	"ErrorCode": 0
	}

2 附录

2.1 附录：群组内“普通消息”与“系统通知”的区别

在群组系统中，APP 管理者向某个群组中的成员推送消息有两个接口，分别是：
1，在某一群组中发送普通消息；
2，在某一群组中发送系统通知。

普通消息与群成员在群内发送的消息等价，其受众只能是全体群成员，且具备漫游能力。系统通知，可以指定消息在群内的受众，但不具备漫游能力。

从应用场景来讲：
1，如果 APP 管理者需要消息具备漫游能力，则应当使用普通消息；
2，如果 APP 管理者需要向群内部分或者全部成员推送一条时效性较高提醒，则应当使用系统通知；
3，如果 APP 需要向某一用户推送消息，且该消息与群组本身没有太大关系，应当使用单发消息接口。