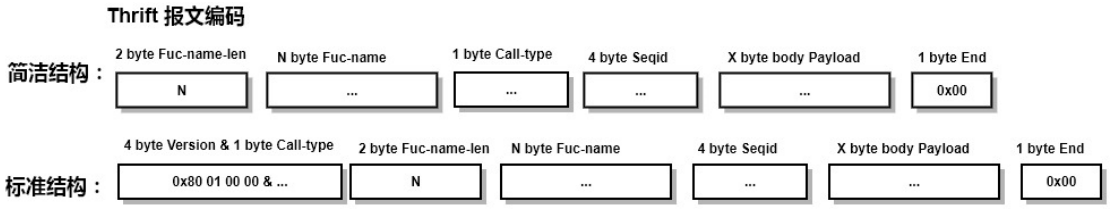
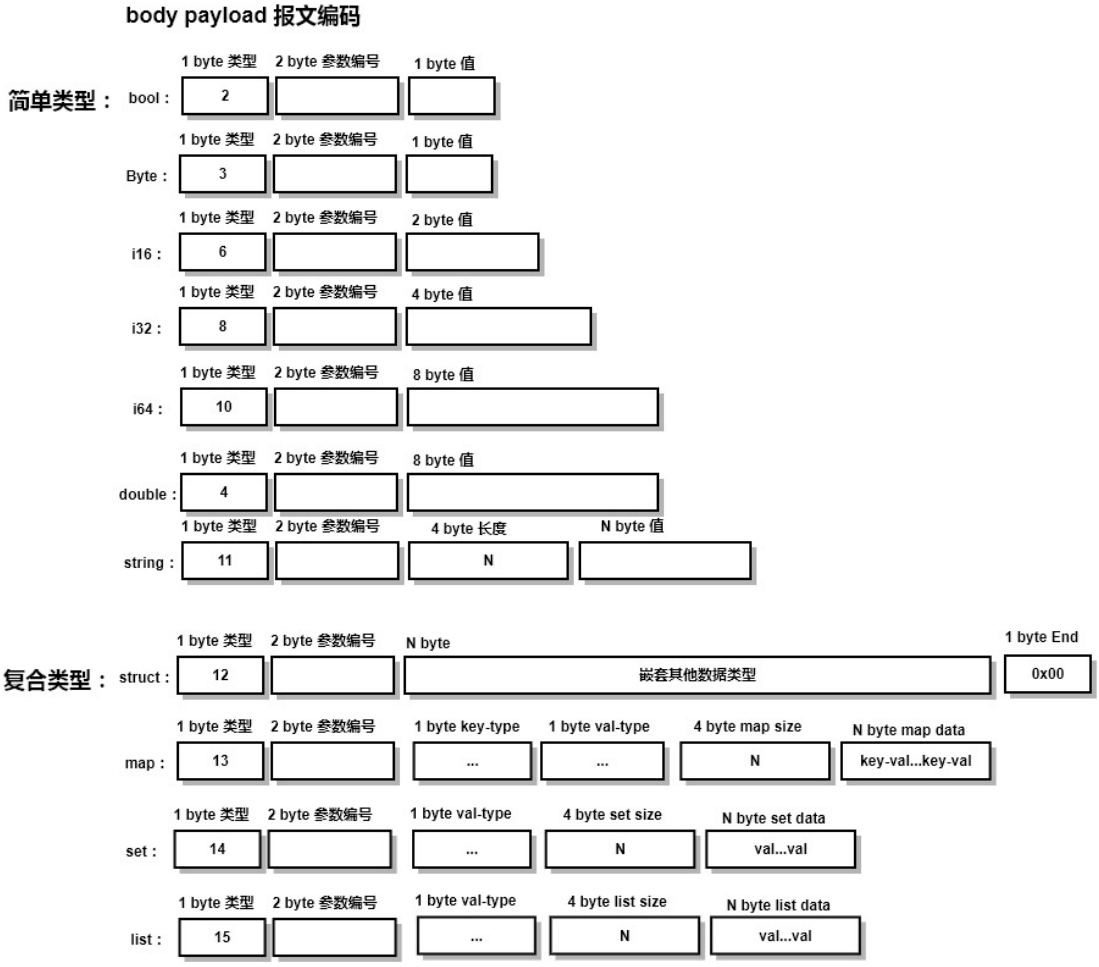
Thrift 二进制编码协议详解

Thrift 报文编码：



body Payload 编码：



示例：

1. **服务定义： getMultipleLoupanLayouts**

```
namespace LoupanService.Loupan
service LoupanService
{
/** 获取一组楼盘的户型列表 */
LayoutResult.LayoutListMapResultDto getMultipleLoupanLayouts(1:
LoupanRequest.GetMultipleLoupanLayoutsRequestDto requestDto)
}
```

2. **参数定义： GetMultipleLoupanLayoutsRequestDto**

```
namespace php LoupanService.Loupan.Request
struct GetMultipleLoupanLayoutsRequestDto
{
    /** 一组楼盘 IDs */
    1: required list<i32> loupanIds

    /** 扩展信息，目前支持：description、partDescriptions */
    2: optional list<string> includables
}
```

3. **返回定义： LayoutListMapResultDto**

```
namespace php LoupanService.Layout.Result
struct LayoutListMapResultDto
{
    1: required i32 code
    2: required string message
    3: optional map<string, string> errors
    4: optional LayoutDto.LayoutListMapDto data
}

## LayoutDto.LayoutListMapDto 定义
namespace php LoupanService.Layout.Dto
struct LayoutDto
{
    /** ID */
    1: required i32 id

    /** 名称 */
    2: required string name

    /** 更新时间 */
```

```
    3: optional string updatedAt
}
typedef list<LayoutDto> LayoutListDto
```

4. 客户端调用

```
use LoupanService\Loupan\LoupanServiceIf;
use LoupanService\Loupan\Request\GetMultipleLoupanLayoutsRequestDto;

// 封装的获取 thrift client 客户端对象
$loupanService =
Apf_Thrift_ThriftServiceFactory::getService(LoupanServiceIf::class);

$requestDto = new GetMultipleLoupanLayoutsRequestDto([
    'loupanIds' => [447101],
]);

$loupan = $loupanService->getMultipleLoupanLayouts($requestDto);
```

5. 剥除传输层协议，抓包请求数据包（返回数据包的解析几乎相同）：

i) ASCII characters

```
evans@kfs-web:~$ od -tc -j 0 /tmp/rrr.rb
0000000 200 001  \0 001  \0  \0  \0 030   g   e   t   M   u   l   t   i
0000020   p   l   e   L   o   u   p   a   n   L   a   y   o   u   t   s
0000040  \0  \0  \0  \0  \f  \0 001 017  \0 001  \b  \0  \0  \0 001  \0
0000060 006 322   }  \0  \0
0000065
```

ii) hexadecimal 1-byte units

```
evans@kfs-web:~$ od -tx1 -j 0 /tmp/rrr.rb
0000000 80 01 00 01 00 00 00 18 67 65 74 4d 75 6c 74 69
0000020 70 6c 65 4c 6f 75 70 61 6e 4c 61 79 6f 75 74 73
0000040 00 00 00 00 0c 00 01 0f 00 01 08 00 00 00 01 00
0000060 06 d2 7d 00 00
0000065
```

6. 详解（网络序为大端）：

0x80010001

= 0x80010000 & 0x01 。代表该消息体为严格的标准结构，且为请求类型（01-请求；02–响应；03-异常；04-oneway）。

0x00000018

= int(24) 。代表后续有 24 个字节为方法名称。

0x67 65 74 4d 75 6c 74 69 70 6c 65 4c 6f 75 70 61 6e 4c 61 79 6f 75 74 73

为方法名称。由第一张 ASCII 对照图可快速得出值为： getMultipleLoupanLayouts

0x00 00 00 00

代表本次消息流水号为 0。

0x0c
= int(12) 。代表该 body Payload 消息类型为 struct 结构体。

0x00 01
表明为第一个参数。

0x0f
= int(15) 。表明该 struct 嵌套了 list 数据类型。

0x00 01
表明为第一个元素。

0x08
表明该 list 里的元素值的数据类型为 i32

0x 00 00 00 01
表明该 list 的元素值的个数为 1。

0x00 06 d2 7d
= int(447101) 。该 list 元素值。

0x00
该 struct 结束标志

0x00
该 thrift 消息结束标志