手机页面和设备通讯分为两种方式：

1. 页面上对web服务器做出请求后，由web服务器发布消息，设备收到消息后，通讯完成。

例如：

在jsp/albumbrowse.jsp中，点击“下载到故事机”按钮后，页面上会发出一个请求

$.ajax({

type : "POST",

url : contextPath + "/track/download.do?deviceId=" + window.deviceId

+ "&id=" + trackListId

+ "&name=" + encodeURI(encodeURI(albumTitle)),

contentType : 'application/json',

async : **true**,

cache:**false**,

//dataType : 'json',

timeout : 7000,

data : JSON.stringify(tracks),

success : **function**(msg) {

console.log(msg);

showAlert("下载成功");

hideMaskLoading();

},

error : **function**(data) {

showAlert("下载失败");

hideMaskLoading();

}

});

使用的api为web api文档中第15点

做出这个请求后，页面就等收到的msg相应，成功或失败，给出相应的提示。

在这里页面没有发布任何消息，但设备仍能收到，这个发布消息的动作是由web服务器执行完**track/download** 后做的。

这种方式的共同点是页面操作的实体是要保存在数据库中的，库结构请参看storybox库结构说明文档，保存完数据后再由web服务器发布消息实现和设备的通信。详细请看web api文档。

1. 页面上直接发布消息，设备收到消息后，通讯完成。

例如：

在jsp/tracklist\_controlbox.jsp中，点击“下一首”按钮后，页面代码如下：

$(".controlboxfooter").find("button.forward").click(**function**(){

forwardTrack();

});

forwardTrack是mqttclient.js中的接口，页面调用此接口，就发布了消息，设备收到这条消息后就执行下一首的操作。

第2种方式的情景较少，目前有指定歌曲播放、上一首、下一首、暂停、继续播放、询问当前播放歌曲、询问当前播放状态、询问当前播放模式、设置列表循环、设置单曲循环、远程关机、设置音量、询问是否在线、询问设备信息、升级指令，详细请看mqttclient.js接口文档。