HTML5技术支持WebApp在手机上拍照，显示在页面上并上传到服务器。这是手机微博应用中常见的功能，当然你也可以在其它类型应用中适当使用此技术。

**1、 视频流**

HTML5 的 The Media Capture（媒体捕捉）API 提供了对摄像头的可编程访问，用户可以直接用 getUserMedia （请注意目前仅Chrome和Opera支持）获得摄像头提供的视频流。我们需要做的是添加一个HTML5的Video 标签，并将从摄像头获得的视频作为这个标签的输入来源。

<video id=”video” autoplay=”"></video>

<script>

var video\_element=document.getElementById(‘video’);

if(navigator.getUserMedia){ // opera应使用opera.getUserMedianow

navigator.getUserMedia(‘video’,success,error);  //success是回调函数,当然你也可以直接在此写一个匿名函数

}

function success(stream){

video\_element.src=stream;

}

</script>  
此时，video标签内将显示动态的摄像视频流。下面需要进行拍照了。

**2、 拍照**

拍照是采用HTML5的Canvas功能，实时捕获Video标签的内容，因为Video元素可以作为Canvas图像的输入，所以这一点很好实现。主要代码如下：

var canvas=document.createElement(‘canvas’); //动态创建画布对象

var ctx=canvas.getContext(’2d’);

var cw=vw,ch=vh;

ctx.fillStyle=”#ffffff”;

ctx.fillRect(0,0,cw,ch);

ctx.drawImage(video\_element,0,0,cw,ch,0,0,vw,vh); //将video对象内指定的区域捕捉绘制到画布上指定的区域，可进行不等大不等位的绘制。

document.body.append(canvas);

**3、 图片获取**

从Canvas获取图片数据的核心思路是用canvas的toDataURL将Canvas的数据转换为base64位编码的PNG图像，类似于“”的格式。

var imgData=canvas.toDataURL(“image/png”);  
这样，imgData变量就存储了一长串的字符数据内容，表示的就是一个PNG图像的base64编码。因为真正的图像数据是base64编码逗号之后的部分，所以要让实际服务器接收的图像数据应该是这部分，我们可以用两种办法来获取。

第一种：是在前端截取22位以后的字符串作为图像数据，例如：

var data=imgData.substr(22);

如果要在上传前获取图片的大小，可以使用：

var length=atob(data).length; //atob 可解码用base-64解码的字串

第二种：是在后端获取传输的数据后用后台语言截取22位以后的字符串（也就是在前台略过上面这步直接上传）。例如PHP里：

$image=base64\_decode(str\_replace(‘data:image/jpeg;base64,’,”,$data);

**4、 图片上传**

在前端可以使用Ajax将上面获得的图片数据上传到后台脚本。例如使用**jQuery**时可以用：

$.post(‘upload.php’,{‘data’:data});

在后台我们用PHP脚本接收数据并存储为图片。

function convert\_data($data){

$image=base64\_decode(str\_replace(‘data:image/jpeg;base64,’,”,$data);

save\_to\_file($image);

}

function save\_to\_file($image){

$fp=fopen($filename,’w');

fwrite($fp,$image);

fclose($fp);

}

以上的解决方案不仅能用于Web App拍照上传，也可以通过Canvas的编辑功能函数提供图片编辑，例如裁剪、上色、涂鸦、圈点等功能，然后把用户编辑完的图片上传保存到服务器上。

在还在不断补充修正的HTML5的驱动下，Web App与Native App之间的距离将越来越小。在可预见的不远的未来，越来越多老的和新的开发项目必将会迁移到WEB应用上来。