返回目录:全网各大厂 iOS 面试题-题集大全

聊一聊 iOS 中的离屏渲染？

GPU 渲染机制：CPU 计算好显示内容提交到 GPU，GPU 渲染完成后将渲染结果放入帧缓冲区，随后视频控制器会按照 VSync 信号逐行读取帧缓冲区的数据，经过可能的数模转换传递给显示器显示。

GPU 屏幕渲染有以下两种方式：

- 1）On-Screen Rendering，意为当前屏幕渲染，指的是 GPU 的渲染操作是在当前用于显示的屏幕缓冲区中进行。
- 2）Off-Screen Rendering，意为离屏渲染，指的是 GPU 在当前屏幕缓冲区以外新开辟一个缓冲区进行渲染操作。

特殊的离屏渲染：如果将不在 GPU 的当前屏幕缓冲区中进行的渲染都称为离屏渲染，那么就还有另一种特殊的“离屏渲染”方式：CPU 渲染。如果我们重写了 drawRect 方法，并且使用任何 Core Graphics 的技术进行了绘制操作，就涉及到了 CPU 渲染。整个渲染过程由 CPU 在 App 内同步地 完成，渲染得到的 bitmap 最后再交由 GPU 用于显示。备注：Core Graphics 通常是线程安全的，所以可以进行异步绘制，显示的时候再放回主线程，一个简单的异步绘制过程大致如下：

```
- (void)display {
 dispatch_async(backgroundQueue, ^{
     CGContextRef ctx = CGBitmapContextCreate(...);
     // draw in context...
     CGImageRef img = CGBitmapContextCreateImage(ctx);
     CFRelease(ctx);
     dispatch_async(mainQueue, ^{
         layer.contents = img;
     });
 });
}
```

离屏渲染的触发方式： * 1）shouldRasterize（光栅化），光栅化是比较特别的一种。光栅化概念：将图转化为一个个栅格组成的图象。光栅化特点：每个元素对应帧缓冲区中的一像素。shouldRasterize = YES 在其他属性触发离屏渲染的同时，会将光栅化后的内容缓存起来，如果对应的 layer 及其 sublayers 没有发生改变，在下一帧的时候可以直接复用。shouldRasterize = YES 这将隐式的创建一个位图，各种阴影遮罩等效果也会保存到位图中并缓存起来，从而减少渲染的频度。相当于光栅化是把 GPU 的操作转到 CPU 上了，生成位图缓存，直接读取复用。当你使用光栅化时，你可以开启 Color Hits Green and Misses Red 来检查该场景下光栅化操作是否是一个好的选择。绿色表示缓存被复用，红色表示缓存在被重复创建。如果光栅化的层变红得太频繁那么光栅化对优化可能没有多少用处。位图缓存从内存中删

除又重新创建得太过频繁，红色表明缓存重建得太迟。可以针对性的选择某个较小而较深的层结构进行光栅化，来尝试减少渲染时间。对于经常变动的内容，这个时候不要开启，否则会造成性能的浪费。例如经常打交道的 TableViewCell，因为 TableViewCell 的重绘是很频繁的（因为 Cell 的复用），如果 Cell 的内容不断变化，则 Cell 需要不断重绘，如果此时设置了 cell.layer 可光栅化，则会造成大量的离屏渲染，降低图形性能。*2）masks（遮罩）*3）shadows（阴影）*4）edge antialiasing（抗锯齿）*5）group opacity（不透明）*6）复杂形状设置圆角等 *7）渐变

为什么会使用离屏渲染：当使用圆角，阴影，遮罩的时候，图层属性的混合体被指定为在未预合成之前（下一个 VSync 信号开始前）不能直接在屏幕中绘制，所以就需要屏幕外渲染被唤起。屏幕外渲染并不意味着软件绘制，但是它意味着图层必须在被显示之前在一个屏幕外上下文中被渲染（不论 CPU 还是 GPU）。所以当使用离屏渲染的时候会很容易造成性能消耗，因为离屏渲染会单独在内存中创建一个屏幕外缓冲区并进行渲染，而屏幕外缓冲区跟当前屏幕缓冲区上下文切换是很耗性能的。由于垂直同步的机制，如果在一个 VSync 时间内，CPU 或者 GPU 没有完成内容提交，则那一帧就会被丢弃，等待下一次机会再显示，而这时显示屏会保留之前的内容不变。这就是界面卡顿的原因。

Instruments 监测离屏渲染：

- 1）Color Offscreen-Rendered Yellow，开启后会把那些需要离屏渲染的图层高亮成黄色，这就意味着黄色图层可能存在性能问题。
- 2）Color Hits Green and Misses Red，如果 shouldRasterize 被设置成 YES，对应的渲染结果会被缓存，如果图层是绿色，就表示这些缓存被复用；如果是红色就表示缓存会被重复创建，这就表示该处存在性能问题了。

iOS 版本上的优化：

- 1）iOS 9.0 之前 UIimageView、UIButton 设置圆角都会触发离屏渲染。
- 2）iOS 9.0 之后 UIButton 设置圆角会触发离屏渲染，而 UIImageView 里 png 图片设置圆角不会触发离屏渲染了，如果设置其他阴影效果之类的还是会触发离屏渲染的。

返回目录:全网各大厂 iOS 面试题-题集大全
