P1:

1. 从事软件开发有6年时间，主要工作集中在桌面软件开发领域。之前从事GIS行业软件开发，这个直观一点就可以理解为开发的是类似百度地图这样的导航软件，现在在网龙office开发部门。
2. 今天就带大家来了解一下桌面软件当中的图形图像处理方法，这些方法的原理都不复杂，但是实现的效果都是实打实的，所谓简约而不简单。

P2:

1. 首先来思考一个问题：你能说出一些什么样的桌面软件。
2. 我们认识的第一款桌面软件很可能是“我的电脑”，exploer也属于一种软件。
3. 简单的比如计算器、记事本这类小工具。当然还有大型商业软件，比如微软Office。
4. 那微软Office或者说“我的电脑”又是怎么来的呢，开发工具软件闪亮登场。
5. 另一大类就是游戏了。
6. 还有呢，我们现在使用的腾讯会议不也正是一种软件吗。它是多平台使用的，大家有些同学估计在使用手机版腾讯会议，那么在手机上运行的腾讯会议可以算做桌面软件吗？我认为如果是广义上的“桌面”软件，这里的桌面带引号，那也可以算。
7. 手机上的“桌面”软件就非常丰富了，比如我们最常使用的拍照软件。
8. 此外，手游也是一大类手机使用者的心头好。
9. 既然我们已经广义上定义了软件了，那么与桌面软件天然势同水火的Web程序呢？不好意思，没有我浏览器Web程序跑都跑不起来。浏览器可是一款新世纪以来最重要的软件啊。

P3:

1. 上面列举了怎么多不同类型的软件，他们形形色色有的用鼠标点点点，有的需要用手指划划划，他们所要完成的任务也南辕北辙，毫不沾边。但是它们有一个共同点，你应该发现了。
2. 对，无论它们内部或者说控制层的逻辑有多么复杂或者多门简单，但是对于使用者来说，或者说对于使用者的屏幕来说，我都一视同仁，总之你要给我画出来。我们说一个人80%以上的信息都是从视觉获得的，哪怕是一个命令行程序，我们也需要它画出一些文字来让我们看。这是软件交互中最终呈现给用户的东西。因此，画的好不好看，酷不酷炫，就决定了一个软件的成败。毕竟，第一眼总是难忘又深刻啊。
3. 好，那么今天我就带大家来看一看，我们要画好一个软件有哪些基本的知识和技巧。

P4:

1. 首先来回顾一下桌面绘制的发展和一些基本背景知识。
2. 早期开发桌面软件的绘制功能以调用系统提供的API来进行。以Windows系统为例Win32一直以来都提供了最基础的绘制API来被调用。Win32的这套API奠定了我们绘制逻辑的基础结构，就是定义了屏幕坐标系、像素点单位、以及一种基本绘制思想，通过组合其提供的基础的点线面图形绘制API来构造复杂绘制逻辑，从而绘制出复杂的图形或图像。
3. 第二阶段随着C++和基于C++的MFC桌面应用开发兴起，早期的绘制API被整合进行了面向对象化，从而有了画布和刷新这两个重要概念，这为构造一些简单的动画效果打下了基础。当然这一时期我们桌面软件的相爱相杀的宿敌Web也是从此出去走了另外一条路，就是进行内核封装，采用CSS这种不一样的绘制体系语法。
4. 好，到了第三阶段，这时我们电脑的硬件环境起来了，绘制不再需要去死抠极限效率问题。所以我们从绘制本身转移到了使用绘制能够实现的业务场景和效果，这个时候C#和Java这些高级语言就为我们封装了更多的绘制工具，使我们能够将注意力集中在业务场景上。这个时候我们的桌面软件绘制也到达了顶峰，上使用像C#的.NET Framework框架和Java AWT或者后期JavaFX框架可以实现几乎所有桌面软件效果。
5. 第四阶段就是必然迎来的3D绘制趋势，这些效果主要就是应用在游戏开发上了。前面三个阶段我们的绘制运算都跑在CPU上，而到了这一阶段，GPU进场。当然，现在GPU已经被比特币玩坏了。不过GPU本身就是为高级绘制运算做服务的。这时期更多就是渲染框架性的项目涌现出来，比如经典的DirectX、OpenGL以及更高级的明星框架虚幻引擎和后来的寒霜引擎。
6. 当然我们常规桌面软件开发不会用到这些效果炸裂的3D或者2D加速渲染，能够掌握像Java这类高级语言的绘制技巧久足以支撑我们开发出足够的桌面软件。

P5:

1. 接下来我们简单回顾下桌面软件绘制的基础知识。
2. 在一个桌面窗口程序中，以左上角作为绘制的原点坐标，横向为x，纵向为y。将绘制区域构造成平面坐标系，这个坐标系的单位有两种：传统绘制模型中的单位是px像素，而在后来Apple推出Retina视网膜屏幕后，引入了dp这个单位。
3. 基本绘制方法包含绘制文字、点线面图形以及图片。
4. 而在代码中，我们将绘制的整个区域抽象为Graphics对象，这个对象一定要依赖于它的device设备。关于设备这个参数还是为历史保留，因为我们知道最早的Win32 API是正经从设备上构造了Graphcis出来，不过后面随着高级语言出现，设备已经被抽象为各种控件了，不过这个历史遗留命名仍然存在。
5. Java中在Graphics上又扩展出Graphics2D，这个对象包含了更丰富一些的绘制方法。下面就是一些常用的绘制方法。
6. 这些绘制方法包括…

P6:目录…

P7:第一节…

P8:我们来看一看渐变、阴影和发光的使用场景…

动态绘制-双缓存：

1. 图像绘制时会经历绘制-背景搽除-绘制的循环过程，假如绘制元素过多绘制时间过长，就会导致明显卡顿从而形成画面闪烁。为了解决这个问题我们引入双缓存或者多缓存技术。而在DirectX中，还用到了交换缓存技术，这样使得渲染速度更快。具体过程为…
2. 双缓存技术的显著特点就在于它使得绘制内容和硬件设备实现了分离…
3. 而过多的需要绘制的元素和较低性能的硬件设备通过双缓存虽然不会造成画面闪烁，但是会造成“跳帧”现象，这就是我们的电脑在高速我们说“该升级配置啦”！