java lsb隐写_lsb隐写详讲

原创

A兰舍硅藻泥 于 2021-03-02 17:23:16 发布 134 收藏 3
文章标签： java lsb隐写

本文链接：https://blog.csdn.net/weixin_42412850/article/details/114971207

那么，怎么隐写呢？

此次给你们讲解的隐写方式即是LSB(最低有效位)隐写。 教程

在前文中，咱们已经介绍了位图，而LSB隐写即是专门针对这种格式的图片的一种隐写方式。 图片

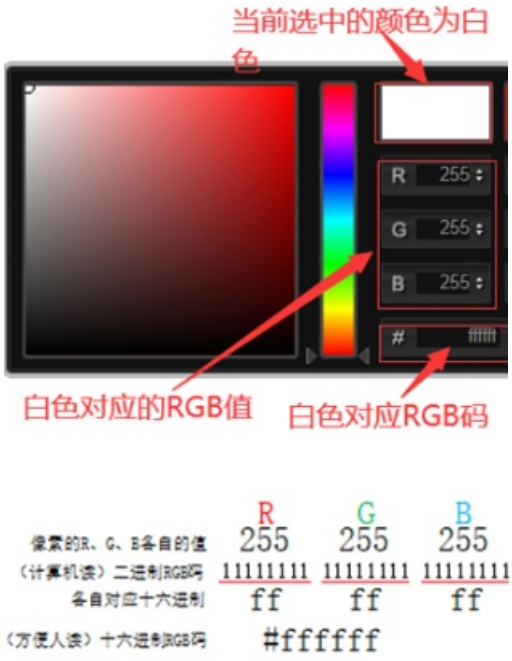
前文中说到位图是由一个个密密麻麻的各类颜色的小方格一行一行的排列而成的精美图片。utf-8

这个小方格，咱们称其为"像素点"。ci

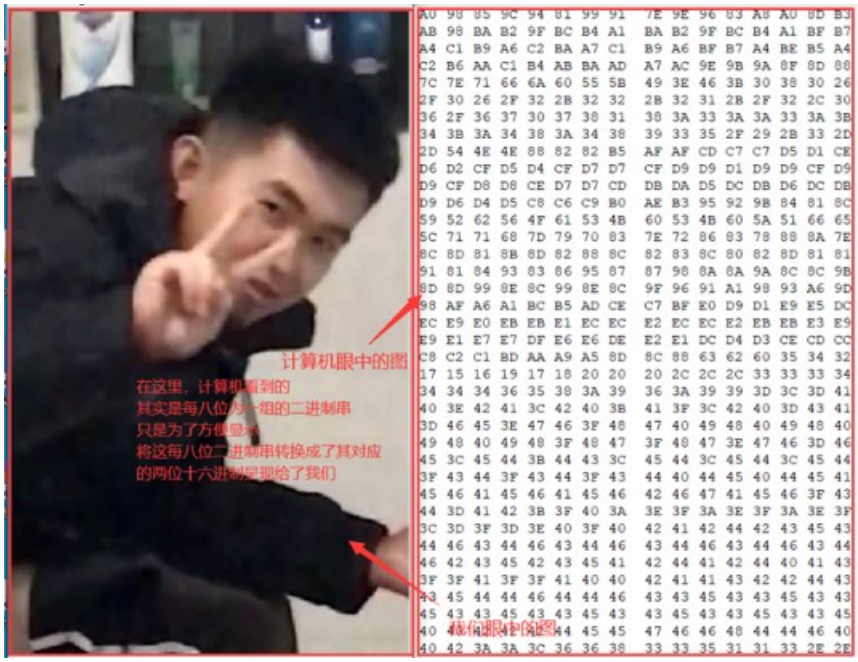
而这些像素点的颜色各类各样才能组成咱们眼前这副彩色的图，那么咱们的计算机是怎么识别变化每一个像素点的颜色的呢？

你们应该都知道红(R)、绿(G)、蓝(B)三元色吧，经过调配这三种颜色，咱们能够获得全部的颜色，而在计算机中，每一个像素点的颜色即是经过调配其R、G、B的所占成分(值)从而获得的，也就是说，每一个颜色的像素点，在计算机看来其实都是一组R、G、B的值。

以下图，咱们选中白色，识别出其R、G、B的值分别为255、255、255。在计算机看来，R、G、B这三种颜色中每一个颜色对应的值都是一个8位二进制数，所以，在计算机读入时，实际上这三元色的值分别为11111111,11111111,11111111,因此，对于计算机而言，它看到的这么一个像素点实际上就是11111111 11111111 11111111这么一个二进制串，咱们称其为该像素点的RGB码(二进制)，为了方便人阅读，咱们人经常将这串二进制串写做十六进制形式，也就是#ffffff，这也是这个像素点的RGB码(十六进制)。



既然对于计算机而言，每一个这样的像素点是一串二进制串(如：11111111 11111111 11111111)，而咱们在前文中已经说到一张位图是由一行一行密密麻麻排列的像素点构成的，那么这么一张图，不就正是一行一行的二进制串嘛！(示意图以下)



那么咱们怎么把"JUST{Guan_xi_ge}"隐写进去呢？

咱们知道，在计算机中，每一个字符其实是用ascii码表示的，那咱们如今把每位字符转换成其对应的八位二进制ascii码形式便可获得该字符串是这样的一串二进制串：

01001010 01010101 01010011 01010100 01111011 01000111 01110101 01100001 01101110 01011111 01111000 01101001 01011111 01100111 01100101 01111101

把咱们要隐写的信息转换成这么一串二进制串后，咱们就开始隐写进这个图片！！！！隐写，正式开始！！！

如今咱们将刚刚转换成的二进制串的每一个二进制位从左至右依次写入上文中所说的图片的每八位二进制的最低位(也就是依次写入图片像素点R、G、B值的最低位)

例如：

按这样写入后，信息的每一位就被咱们隐藏进了像素点的R、G、B对应的8位二进制码的末位(最低位)，所以这种隐写方式被咱们称做"LSB(最低有效位)隐写"

咱们写入信息以后，也看不出图片有多大的变化，为何呢？前面已经说了，在计算机中，每一个像素点的颜色即是经过调配其R、G、B的所占成分(值)从而获得的，那按咱们这种隐写方式，信息写入R、G、B的最低位，若是R、G、B的最低位由于咱们的写入产生变化了，无非也就是1变成了0或者0变成了1，从数值上来说，被咱们隐写过的R、G、B值，最多也就会变化1。这样对每一个像素点说，其R、G、B成分的变化是十分微小的，颜色几乎没变，因此人的肉眼是难以察觉到这样的变化的，所以信息才得以悄声无息地被咱们藏进去！

如下是由python实现的将信息隐藏进guanxi.bmp这副图的python代码实现(注释对重要功能的代码进行了说明，想习得具体函数细节可自行百度(百度：python XXX函数)，"XXX"为你未看懂的函数的名字，必定能搜到的！)：