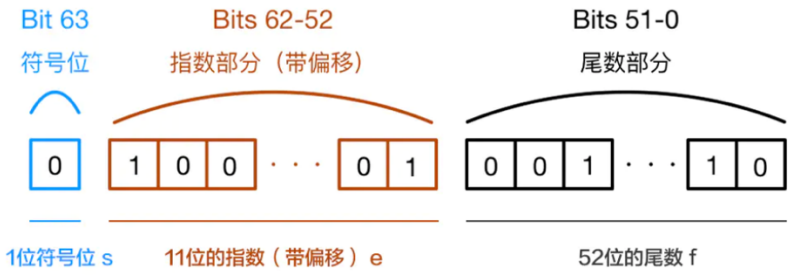
为什么在 JS 里面 0.1+0.2 不等于 0.3？

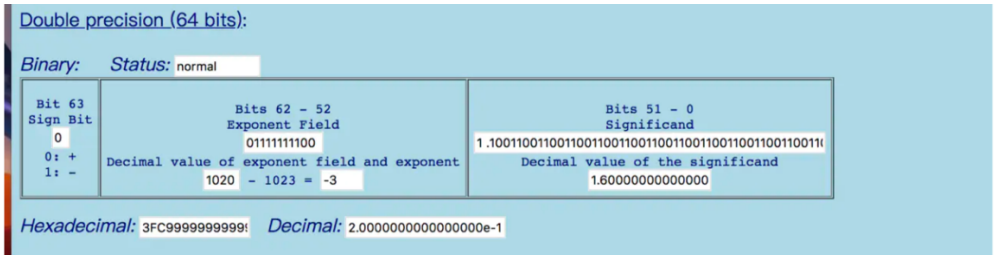
首先先说的我们的结论：精度损失可能出现在进制转换和对接运算过程中

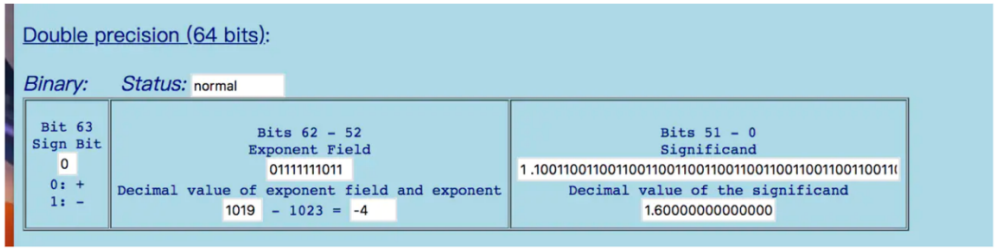
首先我们先谈进制转换：



JS 遵循 IEEE 754 标准，通过 64 位来表示一个数字
Bit 63：符号位，0 表示正数，1 表示负数
Bit 62-52：储存指数部分
Bit 51-0：存储小数部分，既有效数字

那么究竟 0.1 和 0.2 在 IEEE 754 标准内会转化为什么形式呢？





图一为 0.2 的表达，图二为 0.1 的表达
我们可以看到，在转化为二进制的时候，精度已经丢失了，因为 0.1 的正确的二进制表达其实是 0.0001100110011…无限循环，0.2 其实是 0.001100110011…无限循环

我们再来看看对接运算过程：

通过我们上面的得到的已损失精度的二进制数来计算 0.1+0.2，我们得到的最终结果为 0.0100110011001100110011001100110011001100110011001100，转化为十进制就是 0.30000000000000004，并不等于 0.3

综上所述，这就是为什么 0.1+0.2！=0.3

其中一个解决办法是将数字转化为整数：

```
function add(num1, num2) {
 const num1Digits = (num1.toString().split('.')[1] || '').length;
 const num2Digits = (num2.toString().split('.')[1] || '').length;
 const baseNum = Math.pow(10, Math.max(num1Digits, num2Digits));
 return (num1 * baseNum + num2 * baseNum) / baseNum;
}
```

该代码源于网络