二进制数据（包括内存地址）在计算机中一般以16进制的方式表示，大家也习惯使用16进制来查看、记录二进制数据。

MD5信息摘要算法返回的结果是一个128bit的二进制数据，128bit是16个byte，而一个byte转成16进制正好是2位（16进制使用4个bit，一个byte有8个bit），所以MD5算法返回的128bit转成16进制正好是32位。

简单来说，MD5信息摘要之后，返回一个有16个字节的字节数组，而现在需要做的就是把这个字节数组转成16进制格式的数据。

public class Hex {

private static final char *HEXCHARARR*[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};  
  
 private static final String *HEXSTR* = "0123456789abcdef";  
  
 */\*\*  
 \* 字节数组转16进制  
 \*  
 \** ***@param*** *btArr  
 \** ***@return*** *\*/* public static String byteArrToHex(byte[] btArr) {  
 char[] strArr = new char[btArr.length \* 2];  
 int i = 0;  
 for (byte bt : btArr) {  
 strArr[i++] = *HEXCHARARR*[bt >>> 4 & 0xf];  
 strArr[i++] = *HEXCHARARR*[bt & 0xf];  
 }  
 return new String(strArr);  
 }  
  
 */\*\*  
 \* 16进制转字节数组  
 \*  
 \** ***@param*** *hexStr  
 \** ***@return*** *\*/* public static byte[] hexToByteArr(String hexStr) {  
 char[] charArr = hexStr.toCharArray();  
 byte btArr[] = new byte[charArr.length / 2];  
 int index = 0;  
 for (int i = 0; i < charArr.length; i++) {  
 int highBit = *HEXSTR*.indexOf(charArr[i]);  
 int lowBit = *HEXSTR*.indexOf(charArr[++i]);  
 btArr[index] = (byte) (highBit << 4 | lowBit);  
 index++;  
 }  
 return btArr;  
 }  
}

#### byteArrToHex 字节数组转成16进制字符串

基本的算法是：把一个字节分成高4位的bit和低4位的bit，先取高4位的bit，得到这4位bit对应的数字（范围在0~15）,就可以知道对应的16进制数是多少了；按同样的方式取到低4位bit对应的16进制数，把这些16进制数放到一个数组里，最后串成一个字符串，就得到了我们需要的16进制字符串。

#### hexToByteArr 16进制字符串转成字节数组

基本的算法是：把16进制字符串分成一个的char数组，循环取其中的两个char，这两个char的值一定都是在[0123456789abcdef]之间；先找到第一个char对应的下标位置，比如说char的值为 a，那么下标位置就是 10，把数字10转成byte类型，取低4位bit，然后用同样的方式找到第二个char对应的下标数字，取到该数字的低4位bit；将刚才取到的两个低4位bit拼在一起，第一个char对应的4位bit为字节的高4位，第二个char对应的4位bit为字节的低4位，这样组成一个完整的8bit字节；循环完char数组，就得到了16进制字符串对应的字节数组。

因16进制只需占用4个bit，而一个字节有8个bit，所以字节转16进制的时候需要拆开来转换，一个字节对应两个16进制字符；而16进制字符串转成字节的时候，需要用两个16进制数对应的bit拼接起来才能组成一个完整的字节，转换后得到的字节数组的长度是16进制字符串长度的一半。