<https://www.cnblogs.com/fnlingnzb-learner/p/6010165.html>

[复制代码](javascript:void(0);)

1 //1.创建文件夹

2 //import java.io.\*;

3 File myFolderPath = new File(str1);

4 try {

5 if (!myFolderPath.exists()) {

6 myFolderPath.mkdir();

7 }

8 }

9 catch (Exception e) {

10 System.out.println("新建目录操作出错");

11 e.printStackTrace();

12 }

13

14 //2.创建文件

15 //import java.io.\*;

16 File myFilePath = new File(str1);

17 try {

18 if (!myFilePath.exists()) {

19 myFilePath.createNewFile();

20 }

21 FileWriter resultFile = new FileWriter(myFilePath);

22 PrintWriter myFile = new PrintWriter(resultFile);

23 myFile.println(str2);

24 resultFile.close();

25 }

26 catch (Exception e) {

27 System.out.println("新建文件操作出错");

28 e.printStackTrace();

29 }

30

31 //3.删除文件

32 //import java.io.\*;

33 File myDelFile = new File(str1);

34 try {

35 myDelFile.delete();

36 }

37 catch (Exception e) {

38 System.out.println("删除文件操作出错");

39 e.printStackTrace();

40 }

41

42 //4.删除文件夹

43 //import java.io.\*;

44 File delFolderPath = new File(str1);

45 try {

46 delFolderPath.delete(); //删除空文件夹

47 }

48 catch (Exception e) {

49 System.out.println("删除文件夹操作出错");

50 e.printStackTrace();

51 }

52

53 //5.删除一个文件下夹所有的文件夹

54 //import java.io.\*;

55 File delfile=new File(str1);

56 File[] files=delfile.listFiles();

57 for(int i=0;i<files.length;i++){

58 if(files[i].isDirectory()){

59 files[i].delete();

60 }

61 }

62

63 //6.清空文件夹

64 //import java.io.\*;

65 File delfilefolder=new File(str1);

66 try {

67 if (!delfilefolder.exists()) {

68 delfilefolder.delete();

69 }

70 delfilefolder.mkdir();

71 }

72 catch (Exception e) {

73 System.out.println("清空目录操作出错");

74 e.printStackTrace();

75 }

76

77 //7.读取文件

78 //import java.io.\*;

79 // 逐行读取数据

80 FileReader fr = new FileReader(str1);

81 BufferedReader br = new BufferedReader(fr);

82 String str2 = br.readLine();

83 while (str2 != null) {

84 str3

85 str2 = br.readLine();

86 }

87 br.close();

88 fr.close();

89

90 //8.写入文件

91 //import java.io.\*;

92 // 将数据写入文件

93 try {

94 FileWriter fw = new FileWriter(str1);

95 fw.write(str2);

96 fw.flush();

97 fw.close();

98 } catch (IOException e) {

99 e.printStackTrace();

100 }

101

102 //9.写入随机文件

103 //import java.io.\*;

104 try {

105 RandomAcessFile logFile=new RandomAcessFile(str1,"rw");

106 long lg=logFile.length();

107 logFile.seek(str2);

108 logFile.writeByte(str3);

109 }catch(IOException ioe){

110 System.out.println("无法写入文件："+ioe.getMessage());

111 }

112

113 //10.读取文件属性

114 //import java.io.\*;

115 // 文件属性的取得

116 File f = new File(str1);

117 if (af.exists()) {

118 System.out.println(f.getName() + "的属性如下： 文件长度为：" + f.length());

119 System.out.println(f.isFile() ? "是文件" : "不是文件");

120 System.out.println(f.isDirectory() ? "是目录" : "不是目录");

121 System.out.println(f.canRead() ? "可读取" : "不");

122 System.out.println(f.canWrite() ? "是隐藏文件" : "");

123 System.out.println("文件夹的最后修改日期为：" + new Date(f.lastModified()));

124 } else {

125 System.out.println(f.getName() + "的属性如下：");

126 System.out.println(f.isFile() ? "是文件" : "不是文件");

127 System.out.println(f.isDirectory() ? "是目录" : "不是目录");

128 System.out.println(f.canRead() ? "可读取" : "不");

129 System.out.println(f.canWrite() ? "是隐藏文件" : "");

130 System.out.println("文件的最后修改日期为：" + new Date(f.lastModified()));

131 }

132 if(f.canRead()){

133 str2

134 }

135 if(f.canWrite()){

136 str3

137 }

138

139 //11.写入属性

140 //import java.io.\*;

141 File filereadonly=new File(str1);

142 try {

143 boolean b=filereadonly.setReadOnly();

144 }

145 catch (Exception e) {

146 System.out.println("拒绝写访问："+e.printStackTrace());

147 }

148

149 //12.枚举一个文件夹中的所有文件

150 //import java.io.\*;

151 //import java.util.\*;

152 LinkedList<String> folderList = new LinkedList<String>();

153 folderList.add(str1);

154 while (folderList.size() > 0) {

155 File file = new File(folderList.peek());

156 folderList.removeLast();

157 File[] files = file.listFiles();

158 ArrayList<File> fileList = new ArrayList<File>();

159 for (int i = 0; i < files.length; i++) {

160 if (files[i].isDirectory()) {

161 folderList.add(files[i].getPath());

162 } else {

163 fileList.add(files[i]);

164 }

165 }

166 for (File f : fileList) {

167 str2=f.getAbsoluteFile();

168 str3

169 }

170 }

171

172 //13.复制文件夹

173 //import java.io.\*;

174 //import java.util.\*;

175 LinkedList<String> folderList = new LinkedList<String>();

176 folderList.add(str1);

177 LinkedList<String> folderList2 = new LinkedList<String>();

178 folderList2.add(str2+ str1.substring(str1.lastIndexOf("\\")));

179 while (folderList.size() > 0) {

180 (new File(folderList2.peek())).mkdirs(); // 如果文件夹不存在 则建立新文件夹

181 File folders = new File(folderList.peek());

182 String[] file = folders.list();

183 File temp = null;

184 try {

185 for (int i = 0; i < file.length; i++) {

186 if (folderList.peek().endsWith(File.separator)) {

187 temp = new File(folderList.peek() + File.separator

188 + file[i]);

189 } else {

190 temp = new File(folderList.peek() + File.separator + file[i]);

191 }

192 if (temp.isFile()) {

193 FileInputStream input = new FileInputStream(temp);

194 FileOutputStream output = new FileOutputStream(

195 folderList2.peek() + File.separator + (temp.getName()).toString());

196 byte[] b = new byte[5120];

197 int len;

198 while ((len = input.read(b)) != -1) {

199 output.write(b, 0, len);

200 }

201 output.flush();

202 output.close();

203 input.close();

204 }

205 if (temp.isDirectory()) {// 如果是子文件夹

206 for (File f : temp.listFiles()) {

207 if (f.isDirectory()) {

208 folderList.add(f.getPath());

209 folderList2.add(folderList2.peek()

210 + File.separator + f.getName());

211 }

212 }

213 }

214 }

215 } catch (Exception e) {

216 //System.out.println("复制整个文件夹内容操作出错");

217 e.printStackTrace();

218 }

219 folderList.removeFirst();

220 folderList2.removeFirst();

221 }

222

223 //14.复制一个文件夹下所有的文件夹到另一个文件夹下

224 //import java.io.\*;

225 //import java.util.\*;

226 File copyfolders=new File(str1);

227 File[] copyfoldersList=copyfolders.listFiles();

228 for(int k=0;k<copyfoldersList.length;k++){

229 if(copyfoldersList[k].isDirectory()){

230 ArrayList<String>folderList=new ArrayList<String>();

231 folderList.add(copyfoldersList[k].getPath());

232 ArrayList<String>folderList2=new ArrayList<String>();

233 folderList2.add(str2+"/"+copyfoldersList[k].getName());

234 for(int j=0;j<folderList.length;j++){

235 (new File(folderList2.get(j))).mkdirs(); //如果文件夹不存在 则建立新文件夹

236 File folders=new File(folderList.get(j));

237 String[] file=folders.list();

238 File temp=null;

239 try {

240 for (int i = 0; i < file.length; i++) {

241 if(folderList.get(j).endsWith(File.separator)){

242 temp=new File(folderList.get(j)+"/"+file[i]);

243 } else {

244 temp=new File(folderList.get(j)+"/"+File.separator+file[i]);

245 }

246 FileInputStream input = new FileInputStream(temp);

247 if(temp.isFile()){

248 FileInputStream input = new FileInputStream(temp);

249 FileOutputStream output = new FileOutputStream(folderList2.get(j) + "/" + (temp.getName()).toString());

250 byte[] b = new byte[5120];

251 int len;

252 while ( (len = input.read(b)) != -1) {

253 output.write(b, 0, len);

254 }

255 output.flush();

256 output.close();

257 input.close();

258 }

259 if(temp.isDirectory()){//如果是子文件夹

260 folderList.add(folderList.get(j)+"/"+file[i]);

261 folderList2.add(folderList2.get(j)+"/"+file[i]);

262 }

263 }

264 }

265 catch (Exception e) {

266 System.out.println("复制整个文件夹内容操作出错");

267 e.printStackTrace();

268 }

269 }

270 }

271 }

272

273 //15.移动文件夹

274 //import java.io.\*;

275 //import java.util.\*;

276 LinkedList<String> folderList = new LinkedList<String>();

277 folderList.add(str1);

278 LinkedList<String> folderList2 = new LinkedList<String>();

279 folderList2.add(str2 + str1.substring(str1.lastIndexOf("\\")));

280 while (folderList.size() > 0) {

281 (new File(folderList2.peek())).mkdirs(); // 如果文件夹不存在 则建立新文件夹

282 File folders = new File(folderList.peek());

283 String[] file = folders.list();

284 File temp = null;

285 try {

286 for (int i = 0; i < file.length; i++) {

287 if (folderList.peek().endsWith(File.separator)) {

288 temp = new File(folderList.peek() + File.separator + file[i]);

289 } else {

290 temp = new File(folderList.peek() + File.separator + file[i]);

291 }

292 if (temp.isFile()) {

293 FileInputStream input = new FileInputStream(temp);

294 FileOutputStream output = new FileOutputStream(

295 folderList2.peek() + File.separator + (temp.getName()).toString());

296 byte[] b = new byte[5120];

297 int len;

298 while ((len = input.read(b)) != -1) {

299 output.write(b, 0, len);

300 }

301 output.flush();

302 output.close();

303 input.close();

304 if (!temp.delete())

305 System.out.println("删除单个文件操作出错!");

306 }

307 if (temp.isDirectory()) {// 如果是子文件夹

308 for (File f : temp.listFiles()) {

309 if (f.isDirectory()) {

310 folderList.add(f.getPath());

311 folderList2.add(folderList2.peek() + File.separator + f.getName());

312 }

313 }

314 }

315 }

316 } catch (Exception e) {

317 // System.out.println("复制整个文件夹内容操作出错");

318 e.printStackTrace();

319 }

320 folderList.removeFirst();

321 folderList2.removeFirst();

322 }

323 File f = new File(str1);

324 if (!f.delete()) {

325 for (File file : f.listFiles()) {

326 if (file.list().length == 0) {

327 System.out.println(file.getPath());

328 file.delete();

329 }

330 }

331 }

332 //16.移动一个文件夹下所有的文件夹到另一个目录下

333 //import java.io.\*;

334 //import java.util.\*;

335 File movefolders=new File(str1);

336 File[] movefoldersList=movefolders.listFiles();

337 for(int k=0;k<movefoldersList.length;k++){

338 if(movefoldersList[k].isDirectory()){

339 ArrayList<String>folderList=new ArrayList<String>();

340 folderList.add(movefoldersList[k].getPath());

341 ArrayList<String>folderList2=new ArrayList<String>();

342 folderList2.add(str2+"/"+movefoldersList[k].getName());

343 for(int j=0;j<folderList.length;j++){

344 (new File(folderList2.get(j))).mkdirs(); //如果文件夹不存在 则建立新文件夹

345 File folders=new File(folderList.get(j));

346 String[] file=folders.list();

347 File temp=null;

348 try {

349 for (int i = 0; i < file.length; i++) {

350 if(folderList.get(j).endsWith(File.separator)){

351 temp=new File(folderList.get(j)+"/"+file[i]);

352 }

353 else{

354 temp=new File(folderList.get(j)+"/"+File.separator+file[i]);

355 }

356 FileInputStream input = new FileInputStream(temp);

357 if(temp.isFile()){

358 FileInputStream input = new FileInputStream(temp);

359 FileOutputStream output = new FileOutputStream(folderList2.get(j) + "/" + (temp.getName()).toString());

360 byte[] b = new byte[5120];

361 int len;

362 while ( (len = input.read(b)) != -1) {

363 output.write(b, 0, len);

364 }

365 output.flush();

366 output.close();

367 input.close();

368 temp.delete();

369 }

370 if(temp.isDirectory()){//如果是子文件夹

371 folderList.add(folderList.get(j)+"/"+file[i]);

372 folderList2.add(folderList2.get(j)+"/"+file[i]);

373 }

374 }

375 }

376 catch (Exception e) {

377 System.out.println("复制整个文件夹内容操作出错");

378 e.printStackTrace();

379 }

380 }

381 movefoldersList[k].delete();

382 }

383 }

384

385 //17.以一个文件夹的框架在另一个目录创建文件夹和空文件

386 //import java.io.\*;

387 //import java.util.\*;

388 boolean b=false;//不创建空文件

389 ArrayList<String>folderList=new ArrayList<String>();

390 folderList.add(str1);

391 ArrayList<String>folderList2=new ArrayList<String>();

392 folderList2.add(str2);

393 for(int j=0;j<folderList.length;j++){

394 (new File(folderList2.get(j))).mkdirs(); //如果文件夹不存在 则建立新文件夹

395 File folders=new File(folderList.get(j));

396 String[] file=folders.list();

397 File temp=null;

398 try {

399 for (int i = 0; i < file.length; i++) {

400 if(folderList.get(j).endsWith(File.separator)){

401 temp=new File(folderList.get(j)+"/"+file[i]);

402 }

403 else{

404 temp=new File(folderList.get(j)+"/"+File.separator+file[i]);

405 }

406 FileInputStream input = new FileInputStream(temp);

407 if(temp.isFile()){

408 if (b) temp.createNewFile();

409 }

410 if(temp.isDirectory()){//如果是子文件夹

411 folderList.add(folderList.get(j)+"/"+file[i]);

412 folderList2.add(folderList2.get(j)+"/"+file[i]);

413 }

414 }

415 }

416 catch (Exception e) {

417 System.out.println("复制整个文件夹内容操作出错");

418 e.printStackTrace();

419 }

420 }

421

422 //18.复制文件

423 //import java.io.\*;

424 int bytesum = 0;

425 int byteread = 0;

426 File oldfile = new File(str1);

427 try {

428 if (oldfile.exists()) { //文件存在时

429 FileInputStream inStream = new FileInputStream(oldfile); //读入原文件

430 FileOutputStream fs = new FileOutputStream(new File(str2,oldfile.getName()));

431 byte[] buffer = new byte[5120];

432 int length;

433 while ( (byteread = inStream.read(buffer)) != -1) {

434 bytesum += byteread; //字节数 文件大小

435 System.out.println(bytesum);

436 fs.write(buffer, 0, byteread);

437 }

438 inStream.close();

439 }

440 }

441 catch (Exception e) {

442 System.out.println("复制单个文件操作出错");

443 e.printStackTrace();

444 }

445

446 //19.复制一个文件夹下所有的文件到另一个目录

447 //import java.io.\*;

448 File copyfiles=new File(str1);

449 File[] files=copyfiles.listFiles();

450 for(int i=0;i<files.length;i++){

451 if(!files[i].isDirectory()){

452 int bytesum = 0;

453 int byteread = 0;

454 try {

455 InputStream inStream = new FileInputStream(files[i]); //读入原文件

456 FileOutputStream fs = new FileOutputStream(new File(str2,files[i].getName());

457 byte[] buffer = new byte[5120];

458 int length;

459 while ( (byteread = inStream.read(buffer)) != -1) {

460 bytesum += byteread; //字节数 文件大小

461 System.out.println(bytesum);

462 fs.write(buffer, 0, byteread);

463 }

464 inStream.close();

465 } catch (Exception e) {

466 System.out.println("复制单个文件操作出错");

467 e.printStackTrace();

468 }

469 }

470 }

471

472 //提取扩展名

473 String str2=str1.substring(str1.lastIndexOf(".")+1);