آموزش زبان برنامه نویسی جاوا

(کار با فایل) کلاس InputStreamReader

جلسه سی و یکم

نویسنده : رحمان زارعی

جاوا را ساده،آسان و شیرین بنوشید!!!!

در جلسه سی ام به برخی از کلاس های خواندن و نوشتن در زمینه کار با فایل در جاوا پرداختیم. اما خب کار با فایل به همین کلاس ها ختم نمی شود و همان طور که از قبل گفتیم مبحث کار با فایل خیلی گسترده است که ما سعی می کنیم مفاهیم پرکاربردی که بیشتر نیاز بهشون داریم رو کار کنیم.

نکته: برای این که آموزش ها خسته کننده و گیج کننده نشود تکه تکه مفاهیم و کلاس های کار فایل را در جلسات مجزا بررسی می کنیم.

InputStreamReader

کلاس InputStreamReader نیز برای خواندن داده های یک فایل استفاده می شود.

کلاس InputStreamReader پلی از جریان بایت ها به جریان کاراکترهاست. این کلاس داده های بایتی رو میخونه ،برامون این بایت ها را بصورت مجموعه ای از کاراکترها رمزگشایی میکنه و در نهایت ما می توانیم از این مجموعه ای از کاراکترها استفاده کنیم.

برای استفاده از کلاس InputStreamReader باید پکیج java.io.InputStreamReader را درسورس کد برنامه مون import کنیم.

```
import java.io.InputStreamReader ;
```

پس درکل برای خواندن مجموعه ای از کاراکتر ها از کلاس InputStreamReader استفاده می کنیم.

این کلاس نیز دارای چندین سازنده می باشد که با توجه به نیاز آنها را هنگام شی سازی صدا می زنیم:

```
InputStreamReader(InputStream in)
```

یکی از سازنده های این کلاس بصورت پیشفرض برای استفاده از مجموعه ای کاراکترها می باشد.

همون طور که مشاهده می کنید این سازنده به عنوان پارامتر یک شی از کلاس InputStream دریافت می کند.از آنجایی که کلاس FileInputStream فرزند کلاس InputStream هستش این سازنده می تواند یک شی از نوع FileInputStream را نیز دریافت کند.

برخی از متدهای کلاس InputStreamReader را در زیر بررسی کرده ایم:

void close()

این متد برای بستن فایلی است که از طریق کلاس InputStreamReader خوانده شده است.

مثال:

```
package www.javapro.ir;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStreamReader;

public class InputStreamReaderDemo {
	public static void main(String[] args) throws IOException {
		FileInputStream fis = null;
		InputStreamReader isr = null;
		int i;
		char c;

		try {
			// new input stream reader is created
			fis = new FileInputStream("test.txt");
			isr = new InputStreamReader(fis);

			// input stream reader is closed
			isr.close();
			System.out.print("close() invoked ");

			// read() called after closed method
			i = isr.read();
			c = (char) i;
			System.out.println(c);

		} catch (Exception e) {

			// print error
			System.out.print("The stream is already closed");
		} finally {

			// closes the stream and releases resources associated
			if (fis != null)
				fis.close();
			if (isr != null)
				isr.close();
		}
	}
}
```

خروجی: یک استثنا رخ می دهد و دستور درون بلوک catch اجرا می شود،دلیل رخ دادن استثنا این هست که از طریق شی ساخته شده از کلاس InputStreamReader با نام isr متد read صدا زده شده است.و این در حالی است که در خط قبل از آن شی isr که حاوی آدرس فایل خوانده شده از کامپیوتر می باشد با متد close بسته شده است و دیگر نمی توانیم به داده های درون آن دسرسی پیدا کنیم. (چیزی شبیه یک فایل txt در notepad زمانی می توانید که داده های یک فایل تکس را ببینید که آن فایل را باز کرده باشید)

```
close() invoked The stream is already closed
```

```
FileInputStream fis = null;
InputStreamReader isr = null;
```

- از دو کلاس FileInputStream و InputStreamReader بدون صدا زدن سازنده آنها شی ایجاد کرده ایم و مقدار null بهشون نسبت داده ایم. نکته ای که باید به ان توجه کنید این است که به همراه کلاس InputStreamReader حتما باید کلاس FileInputStream هم تعریف کنید چون ورودی پارامتر سازنده کلاس InputStreamReader از نوع کلاس InputStream می باشد.

```
 int i;
char c;
```

- تعریف دو متغیر ، برای داده های بایتی یک متغیر از نوع int و برای داده های کاراکتری یک متغیر از نوع char ایجاد کرده ایم.

```
try {
                // new input stream reader is created
                fis = new FileInputStream("test.txt");
                isr = new InputStreamReader(fis);
```

- برای کنترل استثنا از بلوک try-catch استفاده شده است.
- خب سازنده های دو کلاس را برای اشیا صدا میزنیم.
- آدرس فایلی که از قبل درون کامپیوتر ما و در پوشه پروژه ای که این سورس کد درونش قرار دارد را به عنوان پارامتر به سازنده کلاس FileInputStream داده ایم.
- شی fis که از نوع کلاس FileInputStream هستش را به عنوان پارامتر به سازنده کلاس InputStreamReader داده ایم.

```
isr.close();
```

- شی isr که حاوی جریان فایل ورودی ما می باشد را با این متد بسته ایم.با این کار دیگر فایلی که از کامپیوتر خواندیم بسته شده و دیگر به ان دسرسی نداریم.

```
i = isr.read();
                c = (char) i;
                System.out.println(c);
```

- قصد داشتیم یک بایت را از فایل بخوانیم و با تبدیل بایت خوانده شده به کاراکتر آن را چاپ کنیم که با استثنا روبرو می شویم-زیرا شی isr که حاوی جریان فایل خوانده شده است که در خط با متد close بسته شده است.

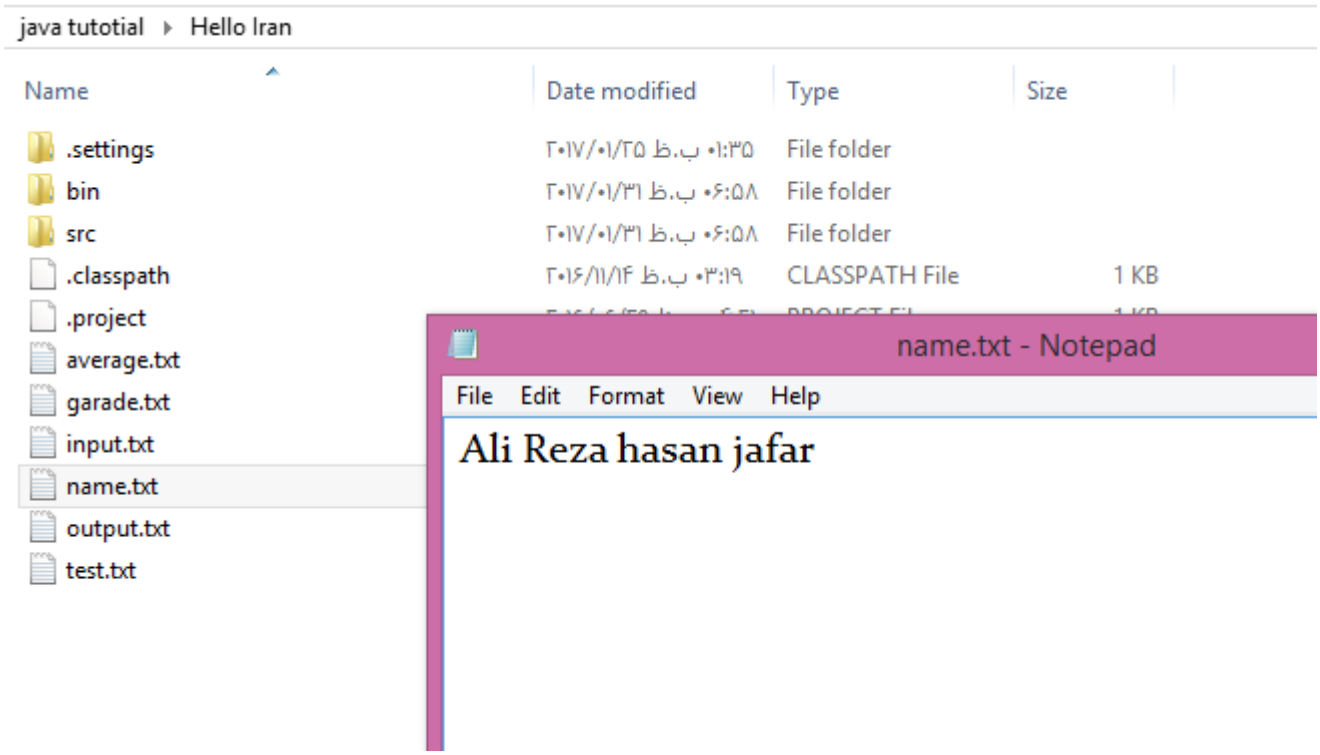
```
} catch (Exception e) {

                // print error
                System.out.print("The stream is already closed");
          } finally {

                // closes the stream and releases resources associated
                if (fis != null)
                      fis.close();
                if (isr != null)
                      isr.close();
          }
```

به دلیل رخ دادن استثنا دستور درون بلوک catch اجرا می شود و در نهایت صرف نظر از این که چه اتفاقی در برنامه رخ می دهد دستورات درون بلوک finally اجرا می شود و درصورت null نبودن فایل ها همه ی آنها بسته می شود.

int read()

این متد تک به تک کاراکترهای موجود در فایل را میخواند.

مثال :در کامپیوتر پوشه پروژمون یک فایل با نام و فرمت name.txt داریم که حاوی تعدادی اسم به صورت متن (رشته) یا مجموعه ای از کاراکتر ها می باشد بصورت زیر: (تصویر ۱) حالا قصد داریم با استفاده از کلاس های FileInputStream و InputStreamReader این فایل را کاراکتر به کاراکتر بخوانیم و در محیط کنسول آنها را چاپ کنیم:



تصویر(۱)

```java
package fileIO;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStreamReader;

public class InputStreamReaderDemo {
   public static void main(String[] args) throws IOException {
      FileInputStream fis = null;
      InputStreamReader isr =null;
      char c;
      int i;

      try {
         // new input stream reader is created
         fis = new FileInputStream("name.txt");
         isr = new InputStreamReader(fis);

         // read till the end of the file
         while((i=isr.read())!=-1)
         {
            // int to character
            c=(char)i;
```

```
            // print char
            System.out.println("Character Read: "+c);
         }
      } catch (Exception e) {

         // print error
         e.printStackTrace();
      } finally {

         // closes the stream and releases resources associated
         if(fis!=null)
            fis.close();
         if(isr!=null)
            isr.close();
      }
   }
}
```

خروجی:

```
Character Read: A
Character Read: l
Character Read: i
Character Read:
Character Read: R
Character Read: e
Character Read: z
Character Read: a
Character Read:
Character Read: h
Character Read: a
Character Read: s
Character Read: a
Character Read: n
Character Read:
Character Read: j
Character Read: a
Character Read: f
Character Read: a
Character Read: r
Character Read:
```

- همان طور که گفتیم ما کاراکتر به کاراکتر فایل را میخوانیم، حالا اینجا هم می بینید که با هر بار خوانده شده کاراکترهای درون فایل مقادیر آنها در محیط کنسول چاپ شده است.مثلا نام اول ما ali بود که بصورت

a

l

i

خوانده شده و بعد از آن فاصله یا اسپیس را به عنوان یک کاراکتر خوانده است و غیره.

نکته جدید نداره فقط به خط کد زیر نگاه کنید:

```
c=(char)i;
```

در اینجا ما مقدار متغیر از نوع بایت را به متغیر از نوع کاراکتر تبدیل ساخته ایم و درون یک متغیر از نوع کاراکتر ریخته ایم. به این کار عمل casting در جاوا می گوییم خود بحث جدایی است واردش نمی شویم فقط این ذهنیت را داشته باشد اگر خواستید یک متغیر را به متغیر دیگر تبدیل کنیم می توانیم با قرار دادن نوع متغیر درون پرانتز و گذاشتن پرانتز سمت چپ متغیر هدف که سمت راست مساوی قرار دارد آن را تبدیل یا به اصلاح casting می کنیم.

```
c=(char)i;
```

- i متغیر هدف ما که سمت راست علامت مساوی قرار دارد.در اینجا متغیر i از نوع int می باشد.
- (char) نوع متغیری که قصد داریم متغیر هدف را به ان تبدیل کنیم درون پرانتز قرار می دهیم.
- حال پرانتز را سمت چپ متغیر هدف قرار می دهیم.

```
(char)i
```

- در نهایت سمت چپ علامت مساوی متغیری که از نوعی که درون پرانتز وجود دارد را قرار می دهیم.

```
c=(char)i;
```

- در اینجا متغیر c از نوع char می باشد.

نکته : همه متغیر ها رو نمی شود به روش casting به هم تبدیل کرد

```
String  b="100000000";
int a=(int)b;
```

مثلا همچین cast ای رو نمیشه انجام داد در جلسه ای جدا به این مفهوم می پردازیم ،بگذریم فراموشش کنید بریم سراغ اصل مطلب!!!

مثال بالا را بصورت زیر تغییر می دهیم:

```
package fileIO;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStreamReader;
```

```java
public class InputStreamReaderDemo {
   public static void main(String[] args) throws IOException {
      FileInputStream fis = null;
      InputStreamReader isr =null;
      char c;
      int i;

      try {
         // new input stream reader is created
         fis = new FileInputStream("name.txt");
         isr = new InputStreamReader(fis);

         // read till the end of the file
         while((i=isr.read())!=-1)
         {
            // int to character
            c=(char)i;

            // print char
            System.out.print(c);
         }
      } catch (Exception e) {

         // print error
         e.printStackTrace();
      } finally {

         // closes the stream and releases resources associated
         if(fis!=null)
            fis.close();
         if(isr!=null)
            isr.close();
      }
   }
}
```

خروجی:

```
Ali Reza hasan jafar
```

همان طور که مشاهده می کنید با وجود کاراکتر به کاراکتر خواندن فایل "name.txt" ، دقیقا عین متن مورد نظر درون فایل را در کنسول چاپ کردیم.

این دیگه دست شماست که هنگام خواندن داده ها از فایل ، به چه صورت داده ها را دستکاری کنید و غیره

ما کلاس های زیادی در مورد کاربا فایل داریم که پرکاربردترین آنها رو بررسی می کنیم. اصلا هم نگران گسترده بودن مفاهیم کار با فایل در جاوا نباشید چون نیاز به یادگیری همه این مفاهیم نیست!!!! و تنها دو کلاسی که راحت و ساده در جهت خواندن و نوشتن فایل هستند و امکانات و توانایی های بیشتری دارند رو یاد بگیرید کافیه!! سایر کلاس ها هم اگر نیاز پیدا کردید خب به راحتی یاد می گیرید چون همه کلاس های کار با فایل یک الگو و روش مشخص دارند یعنی برخی از کلاس ها در دسته خواندن فایل و برخی دیگر در دسته نوشتن فایل قرار می گیرند.

پیروز و موفق باشید
