برنامه نویسی جاوا

آموزش پروژه محور

نویسنده:رحمان زارعی

جاوا را ساده،آسان و شیرین بنوشید!!!!

نمونه مثال کار با فایل در جاوا سری پنجم

مثال: برنامه ای به زبان جاوا بنویسید که آدرس یک فایل در کامپیوتر را بگیرد، و به آدرسی که بهش می دهیم فایل را در آن محل ایجاد کند. (چیزی شبیه کپی پیست در کامپیوتر که در این برنامه به صورت دستی آدرس مبدا فایل و مقصد فایل به برنامه داده میشه)

پیش نیاز: مباحث کار با فایل در جاوا + مباحث پایه و شی گرایی جاوا

پاسخ:

```
package www.javapro.ir;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Scanner;
public class CopyFiles {
	static void copyFileUsingStream(File source, File dest) {

		try {
			InputStream is = new FileInputStream(source);
			OutputStream os = new FileOutputStream(dest);
			byte[] buffer = new byte[1024];
			int length;
			while ((length = is.read(buffer)) > 0) {
				os.write(buffer, 0, length);
			}
			is.close();
			os.close();
		} catch (IOException e) {
		}

	}
	public static void main(String[] args) {

		String addressSource = "E:\\picture\\gisekan.jpg";

		String addressDestination = "F:\\image\\";
```

```
            File fileSource = new File(addressSource);

            File fileDestination = new File(addressDestination
                        + fileSource.getName());
            copyFileUsingStream(fileSource, fileDestination);
            System.out.println("Succes :-)");
        }

}
```

اصلا نگران نباشید!!! برنامه کپی و پیست کردن یک فایل به سادگی خواندن و نوشتن در فایل هستش که در جلسات آموزشی به آن پرداخته ایم. برای کپی کردن یک فایل ابتدا نیاز است که فایل مبدا را باز کرده و بخوانیم و تمام اطلاعات درون فایل را در یک فایل جدید و خامی که ایجاد کرده ایم. ذخیره سکنیم.

همان طور که جلسات آموزشی مبحث فایل گفتیم ، فایل ها از جریانی از بایت ها تشکیل شده اند، حالا برای کپی کردن یک فایل کافیست این جریان بایت ها را در فایل بخوانیم و یک نسخه از انها رو کپی کنیم و بریزیم داخل یک فایل جدید و خام.

بصورت عملی در زیر کپی و پیست کردن یک فایل را یاد خواهیم گرفت.

توضیحات برنامه:

این برنامه یک متد با نام copyFileUsingStream دارد که از طریق ان می توان کپی از هر نوع فایل را در مکان مورد نظر ایجاد کرد.

متد copyFileUsingStream را بررسی می کنیم:

```
static void copyFileUsingStream(File source, File dest)
```

- در این خط کد متد copyFileUsingStream دو پارامتر به عنوان ورودی میگیرد که هر دو از نوع کلاس File می باشد.
- پارامتر source آدرس محل فایلی است که قراره آن را کپی کنیم.
- پارامتر dest آدرس محلی است که قراره فایل کپی شده ما در آن پیست شود.

```
InputStream is = new FileInputStream(source);
```

- ایجاد یک شی از نوع کلاس InputStream برای خواندن فایل مبدا (فایلی که قرار از روی آن کپی برداریم)

- سازنده ای که بعد از کلمه کلیدی new صدا زده شده است ، سازنده کلاس FileInputStream هستش که فرزند کلاس InputStream می باشد.در اینجا از چندریختی استفاده کرده ایم چرا؟ زیرا دوست داریم به ویژگی ها و رفتار های کلاس InputStream دسرسی پیدا کنیم اما چون کلاس InputStream یک کلاس abstract و انتزاعی هستش نمی توان از ان شی ایجاد کرد به همین خاطر به شیوه چندریختی کاری کردیم که بتوانیم از طریق سازنده فرزندش به ویژگی های و رفتارهای آن دسرسی پیدا کنیم.
- سازنده این کلاس به عنوان پارامتر یک یک شی از نوع کلاس File می گیرد.

```
OutputStream os = new FileOutputStream(dest);
```

- در این خط کد یک فایل خام در آدرس مقصدی که فایل قراره پیست شود ایجاد کرده ایم.اطلاعات فایل مبدا را قراره در این فایل کپی و ذخیره کنیم.

```
byte[] buffer = new byte[1024];
```

- ما قصد داریم بایت به بایت فایل مبدا را بخوانیم ، در اینجا آرایه ای از نوع byte تعریف کرده ایم که بایت های خوانده شده از فایل را درون آن بریزیم.
- در مورد سایز آرایه هم از آنجایی که اندازه بایت های موجود در فایل مبدا ای که قصد داریم بخوانیم را نمی دانیم ،اندازه این ارایه را ۱۰۲۴ انتخاب کردیم، معمولا اندازه این آرایه را ۱۰۲۴ انتخاب می کنند وگرنه اندازه ۵۰۰ هم میشه داد اما دیگه اندازه متداولی که براش قرار می دهند یعنی ۱۰۲۴ را انتخاب کرده ایم.

```
int length;
                while ((length = is.read(buffer)) > 0) {
                     os.write(buffer, 0, length);
                }
```

- length تعداد بایت هایی که در فایل نوشته می شوند.
- read بایت های فایل مبدا را میخواند.
- متد write آرایه buffer که حاوی بایت های خوانده از شده از فایل مبدا است را در فایل مقصد میریزد.
- کاربرد مقدار عدد صفر در متد write(buffer, 0, length) چیست؟ عدد صفر یعنی از اندیس و خانه صفرم (اول) آرایه شروع کن. buffer[0] .

```
     is.close();
                os.close();
```

- در پایان فایل های خود را با متد close می بندیم.

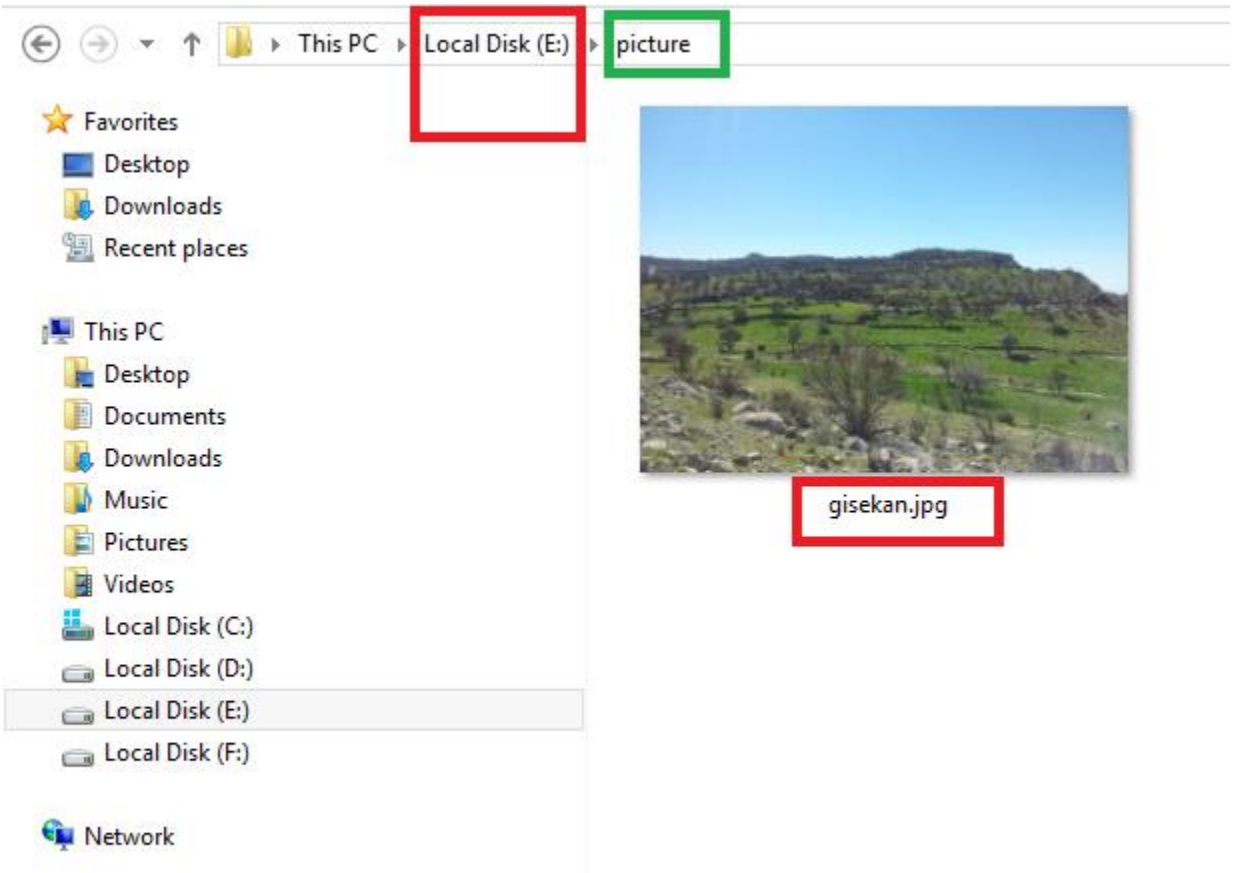
حال برای تست این برنامه در متد main کلاس بصورت زیر عمل کرده ایم:

```
public static void main(String[] args) {
```

- متد main کلاس که برای تست و اجرای برنامه ازش استفاده می کنیم.

```
String addressSource = "E:\\picture\\gisekan.jpg";
```

- در این برنامه ما قصد داریم یک فایل تصویری با نام و فرمت **gisekan.jpg** را در درایو E و پوشه picture را کپی کنیم.
- پس فایلی که در این آدرس قرار دارد فایل مبدا ما را تشکیل می دهد: تصویر(۱)



تصویر(۱)

- ما قصد داریم یک کپی از فایل **gisekan.jpg** را در مکان دیگر از کامپیوتر ایجاد کنیم.

- دقت کنید هنگام آدرس دهی به جای یک علامت "\" از دو علامت "\\" استفاده کنیم.
- نام و فرمت فایلی که قصد داریم از روی ان یک کپی برداریم نیز در آخر آدرس قید کنیم.

```
String addressDestination = "F:\\image\\";
```

- آدرس مکانی که فایل ما قراره در آنجا کپی و پیست شود را درون یک متغیر از نوع رشته می ریزیم.
- این مکان در درایو F و پوشه image می باشد.

```
File fileSource = new File(addressSource);
```

- یک شی از نوع کلاس File ایجاد کرده و درون ان متغیر addressSource که حاوی آدرس فایل مبدا می باشد را قرار داده ایم.

```
File fileDestination = new File(addressDestination
                       + fileSource.getName());
```

- یک شی از نوع کلاس File ایجاد کرده و درون آن متغیر addressDestination که حاوی آدرس مکانی است که فایل ما قراره در آنجا پیست شود (ایجاد شود) قرار گرفته است.
- درون سازنده کلاس File بعد از پارامتر addressDestination یک علامت "+" آورده شده و بعد از علامت جمع دستور زیر آمده است:

```
fileSource.getName();
```

- این دستور نام و فرمت فایل ما را برمی گرداند (به ما می دهد) در این مثال نام و فرمت فایل ما که تصویری است برمی گرداند. در اینجا مقدار "**gisekan.jpg** " که نام و فرمت فایل تصویرمون هست به ما داده می شود.
- اگر توجه کرده باشید ما در متغیر addressDestination تنها آدرس مکانی که فایل قراره در آن کپی شود را ریخته ایم و خبری از نام و فرمت فایل نیست!!! :

```
"F:\\image\\";
```

خب برای این که نام و فرمت فایل هم به این آدرس اضاف کنیم با کمک علامت + و دستور `fileSource.getName()` نام و فرمت فایل را نیز به آدرس اضاف می کنیم. و نتیجه بصورت زیر خواهد بود:

```
F:\image\gisekan.jpg
```

- توجه داشته باشیم ما در کدنویسی بین درایو و پوشه و فایل از دو علامت "\\" استفاده می کنیم و سیستم هنگام خواندن بصورت "\" تشخیص می دهد.

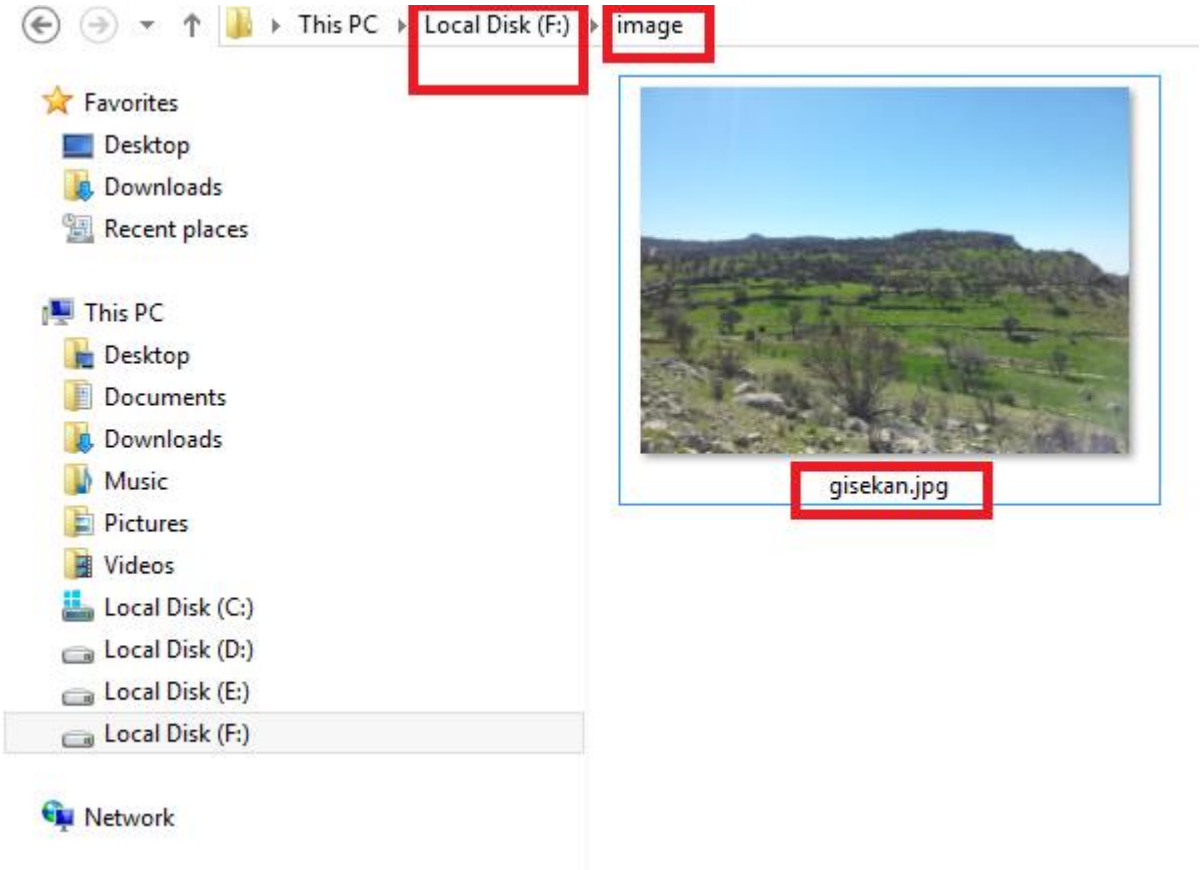
```
copyFileUsingStream(fileSource, fileDestination);
        System.out.println("Succes :-)");
```

- خب متد copyFileUsingStream را برای کپی کردن فایل مبدا درون فایل مقصد را صدا می زنیم.
- متد copyFileUsingStream یک متد static هستش به همین خاطر برای صدا زدن آن در متد main نیاز به شی سازی از کلاس نداریم.
- نتیجه عملیات این است که فایل gisekan.jpg که در آن زیر قرار دارد را:

```
"E:\\picture\\gisekan.jpg";
```

به مکان زیر انتقال می دهد: تصویر(۲)

```
F:\image\gisekan.jpg
```



تصویر(۲)

در پایان هم پیام " success :-)" در کنسول مبنی بر موفق بودن عملیات چاپ می شود.

نکته: ما هر فایل با هر نوع فرمتی را می توانیم از طریق این برنامه در مکان مورد نظر خود در کامپیوتر کپی کنیم.

نظرتون در مورد این نوع سبک آموزش را برای ما ارسال کنید، قطعا نظرات شما در بهبود آموزش های جاوا تاثیر گذار است.

برای دریافت نمونه مثال ها و آموزش جاوا کانال و سایت ما را دنبال کنید.

پیروز و موفق باشید

سایت آموزش زبان جاوا به زبان ساده،آسان و شیرین!!!

www.JAVAPro.ir

آموزش جاوا SE را با تجربه شخصی و به زبان خودمونی یاد بگیرید!!!!!

بازدید از کانال

بازدید از سایت

هر روز مفاهیم و مثال های جدید به سایت اضافه می شود برای اطلاع از مطالب جدید روی سایت عضو کانال شوید.