آموزش زبان برنامه نویسی جاوا

کلاس های داخلی

جلسه نوزدهم

نویسنده : رحمان زارعی

جاوا را ساده،آسان و شیرین بنوشید!!!!

در این جلسه قصد داریم در مورد کلاس های داخلی یا تو در تو در جاوا بحث کنیم!

کلاس های داخلی:

در جاوا درست مثل متدها و متغیرها یک کلاس می تواند یک کلاس دیگر را به عنوان عضو در بدنه خود داشته باشد!

جاوا به شما اجازه می دهد که یک کلاس را در داخل کلاسی دیگر بنویسید.پس به کلاسی که در داخل کلاس دیگر تعریف می شود،کلاس داخلی و کلاسی که کلاس داخلی درون آن هست کلاس بیرونی می گوییم.

نحوه نوشتن کلاس داخلی :

در زیر نحوه نوستن کلاس داخلی را مشاهده میکنید.در اینجا کلاس `Inner_Demo` یک کلاس داخلی

و کلاس `Outer_Demo` یک کلاس بیرونی هست.

```
package javalike;

public class Outer_Demo {

    class Inner_Demo {

    }
}
```

کلاس های داخلی به دو نوع تقسیم می شوند:

کلاس های داخلی استاتیک (Static Inner classes):

- کلاس هایی که اعضای آن استاتیک هستند.

کلاس های داخلی غیر استاتیک (Non-Static Inner classes):

- کلاس هایی که اعضای آن غیر استاتیک هستند.

کلاس های داخلی غیر استاتیک:

کلاس های داخلی یک مکانیزم امنیتی در جاوا هستند.ما می دانیم که modifier یا سطح دسترسی یک کلاس نمی تواند **private** باشد.اما اگر ما از یک کلاس به عنوان عضو دیگری از کلاس(کلاس داخلی) استفاده کنیم می توانیم سطح دسترسی کلاس داخلی را **private** تعریف کنیم. این کار باعث خصوصی سازی اعضای یک کلاس می شود.

کلاس های داخلی بستگی به این که چطور و کجا ازشون استفاده کنیم سه نوع هستند:

۱. کلاس داخلی
۲. کلاس داخلی های محلی درون متد
۳. کلاس های داخلی بی نام.

کلاس داخلی:

ایجاد یک کلاس داخلی خیلی ساده است!شما تنها نیاز دارید یک کلاس را درون کلاس دیگر پیاده سازی! بر خلاف یک کلاس بیرونی یک کلاس داخلی می تواند **private** باشد،هنگامی که شما یک کلاس داخلی را private تعریف میکنید،در خارج از کلاس بیرونی از طریق شی نمی توان به کلاس داخلی و اعضای آن دسترسی پیدا کرد!
برای درک بهتر به دو مثال زیر توجه کنید:

```
package javalike;

class Outer_Demo {

	private class Inner_Demo {

		int a = 5;
	}

	Inner_Demo in = new Inner_Demo();

}

public class Test {

	public static void main(String[] args) {
		Outer_Demo out = new Outer_Demo();
		System.out.println(out.in.a);
	}

}
```

خروجی:

خطای کامپایل!

به خطایی که حین کد زدن در محیط ایکلیپس نمایش داده میشود خطای کامپایل می گوییم.

به خطایی که بعد از اجرای برنامه نمایش داده می شود خطای زمان اجرا می گوییم.

- در این مثال ما سه کلاس داریم که دو کلاس Outer_Demo و Inner_Demo تو در تو هستند. و کلاس Test که دارای متد main می باشد برای اجرای دستورات برنامه ازش استفاده میکنیم.

```
	private class Inner_Demo {
		int a = 5;
	}
```

- در کلاس Outer_Demo یک کلاس داخلی به نام Inner_Demo که از نوع private هستش و یک متغیر صحیح به نام a عضو آن می باشد داریم.

```
Inner_Demo in = new Inner_Demo();
```

- در بدنه کلاس Outer_Demo یک شی از کلاس داخلی Inner_Demo ساخته ایم (تا اینجا مشکلی نیست چون در کلاس بیرونی می توان از کلاس داخلی شی ساخت).

```
public class Test {
	public static void main(String[] args) {
		Outer_Demo out = new Outer_Demo();
		System.out.println(out.in.a);
	}
}
```

- در کلاس Test قصد داریم یک شی از کلاس بیرونی Outer_Demo بسازیم و از طریق آن به شی کلاس داخلی Inner_Demo به نام in و از طریق شی in به متغیر درون کلاس داخلی Inner_Demo دسرسی پیدا کنیم.

- چون کلاس داخلی Inner_Demo از نوع private هستش نمی توان خارج کلاس بیرونی،از طریق شی به ان دسرسی پیدا کرد! تغییرات کد بالا رو در زیر مشاهده کنید:

در مثال زیر که تغییر یافته مثال بالا هست تنها کلاس داخلی Inner_Demo از نوع private نمی باشد! و همان طور که مشاهده میکنید هیچ خطایی در کامپایل و اجرا آن به وجود نیامده است:

```
package javalike;

class Outer_Demo {

	class Inner_Demo {

		int a = 5;
	}

	Inner_Demo in = new Inner_Demo();

}

public class Test {

	public static void main(String[] args) {
		Outer_Demo out = new Outer_Demo();
		System.out.println(out.in.a);
	}

}
```

خروجی:

```
5
```

- پس به این نتیجه میرسیم که اگر یک کلاس داخلی از نوع private تعریف کردیم در خارج از کلاس بیرونی نمی توان از طریق شی به ان دسرسی پیدا کرد.
- در زیر روش دسرسی به یک کلاس داخلی private در خارج از کلاس بیرونی آمده است: (این مثال تغییر یافته مثال اول می باشد)

```
package javalike;

class Outer_Demo {

   private class Inner_Demo {

         int a = 5;
   }

   public void print() {
         Inner_Demo in = new Inner_Demo();
         System.out.println("value of a is " + in.a);

   }
}

public class Test {

   public static void main(String[] args) {
         Outer_Demo out = new Outer_Demo();
         out.print();
   }

}
```

خروجی:

```
value of a is 5
```

- در اینجا کلاس داخلی ما private هستش اما تونستیم به مقدار متغیر صحیح a درون ان دسرسی پیدا کنیم.
- چگونه؟! کلاس داخلی حتی اگر private هم باشه باز می شود در داخل کلاس بیرونی که در آن قرار دارد از ان شی ساخت! پس ما در متد print یک شی از کلاس داخلی Inner_Demo به نام in ساختیم و از طریق شی ساخته شده به اعضای کلاس داخلی دسرسی پیدا کردیم.حال با ساخت شی از کلاس بیرونی در کلاس Test متد print را صدا زدیم و دستور درون آن بدون خطا اجرا شد.

کلاس داخلی محلی – درون متد:

در جاوا ما می توانیم یک کلاس را داخل یک متد و از نوع محلی شبیه متغیرهای محلی تعریف کنیم.دامنه دسرسی به این نوع کلاس محدود به بدنه متد می باشد.پس یک کلاس داخلی محلی تنها در بدنه یک متد قابل تعریف می باشد.

در زیر نحوه پیاده سازی یک کلاس داخلی محلی که درون یک متد تعریف شده است را مشاهده میکنید:

```
package javalike;

public class OuterClass {

	public void methodOuter() {
		class MethodInner_Demo {
			public void print() {

				System.out.println("www.javalike1.ir");

			}
		}
		MethodInner_Demo in = new MethodInner_Demo();
		in.print();
	}

	public static void main(String[] args) {
		OuterClass out = new OuterClass();
		out.methodOuter();
	}
}
```

خروجی:

```
www.javalike1.ir
```

- رنگ سبز محدوده کلاس بیرونی ما می باشد.
- رنگ زرد محدوده کلاس داخلی محلی ما که درون متدی از کلاس بیرونی قرار گرفته است.
- همان طور که مشاهده میکنید ما یک کلاس داخلی محلی به نام `MethodInner_Demo` تعریف کرده ایم که یک متد درون خود داره و یک پیامی چاپ میکند.حال در متد `methodOuter` کلاس بیرونی `OuterClass` یک شی به نام `in` از کلاس داخلی محلی `MethodInner_Demo` ساخته ایم و از طریق شی `in` متد `print` درون کلاس داخلی محلی `MethodInner_Demo` را صدا زده ایم،در پایان یک شی از کلاس بیرونی `OuterClass` ساخته و از

طریق شی ساخته شده از آن متد methodOuter صدا میزنیمفبا این کار دستورات درون این متد اجرا و پیام در خروجی چاپ می شود.

کلاس داخلی پی نام :

در این جلسه زوده بهش بپردازیم! در مباحث شی گرایی و اینترفیس آن را بررسی میکنیم.

کلاس داخلی استاتیک :

یک کلاس داخلی استاتیک یک کلاس تو در تو است که مانند متغیرها و متدهای استاتیک یک کلاس،عضو استاتیک کلاس بیرونی می باشد.از خصوصیات کلاس داخلی استاتیک صدازدن و دسرسی به ان مانند متغیرها و متدهای استاتیک یک کلاس بدون نیاز به نمونه سازی و شی ساختن از کلاسی که درونش قرار دارد می باشد.

نحوه پیاده سازی یک کلاس داخلی استاتیک را ببینید:

```
package javalike;

public class OuterClass {

	static class innerClass {

	}

}
```

❖ کافیست ابتدای یک کلاس داخلی از کلمه کلیدی **static** استفاده کنیم.

نحوه نمونه سازی از کلاس داخلی استایک کمی متفاوت تر از نمونه سازی از کلاس داخلی عادی می باشد.
مثال زیر را ببینید:

```
package javalike;

public class OuterClass {

	static String email = "Rahman.zarie92@gmail.com";

	static class InnerClass {
		int b = 5;

		public void print() {

			System.out.println("@javalike");
		}

	}

	public static void main(String[] args) {
		OuterClass.InnerClass out = new OuterClass.InnerClass();
		out.print();
		System.out.println("value of b " + out.b);
		System.out.println("Email is "+OuterClass.email);

	}
}
```

خروجی:

```
@javalike
value of b 5
Email is Rahman.zarie92@gmail.com
```

- رنگ سبز کلاس بیرونی و رنگ زرد کلاس داخلی استاتیک ما را نشان می دهد.
- کلاس داخلی استاتیک ما دارای یک عضو عدد صحیح و یک متد می باشد.

```
OuterClass.InnerClass out = new OuterClass.InnerClass();
```

- در اینجا نمونه سازی و شی ساختن از کلاس داخلی استاتیک با کلاس داخلی عادی اندکی متفاوت است.همان طور که گفتیم یک کلاس داخلی استاتیک مانند متغیر ها و متدها عضوی از یک کلاس می باشد.اگر یادتون باشه ما برای صدا زدن یک عضو استاتیک کلاس از نام کلاس و نقطه استفاده می کردیم . در این جا متغیر رشته email از نوع استاتیک و در بدنه کلاس بیرونی ما می باشد،روش صدا زدن آن در متدmain را ببینید:

```
System.out.println("Email is "+OuterClass.email);
```

```
className+.+menbersClass
```

پس متغیر یا متدی که در بدنه کلاس استاتیک باشد تنها با نام کلاس +نقطه +عضو موردنظر قابل دسرس می باشد.حال یک کلاس داخلی نیز عضوی از کلاس است و از جهتی استاتیک هم میباشد پس به راحتی برای نمونه سازی از ان از طریق نام کلاس بیرونی+نقطه+نام کلاس داخلی استاتیک استفاده میکنیم

نام شی+ نام کلاس داخلی استایک + . +نام کلاس بیرونی

```
OuterClass.InnerClass out
```

و بعدش علامت مساوی و سمت راست علامت مساوی بصورت زیر عمل میکنیم:

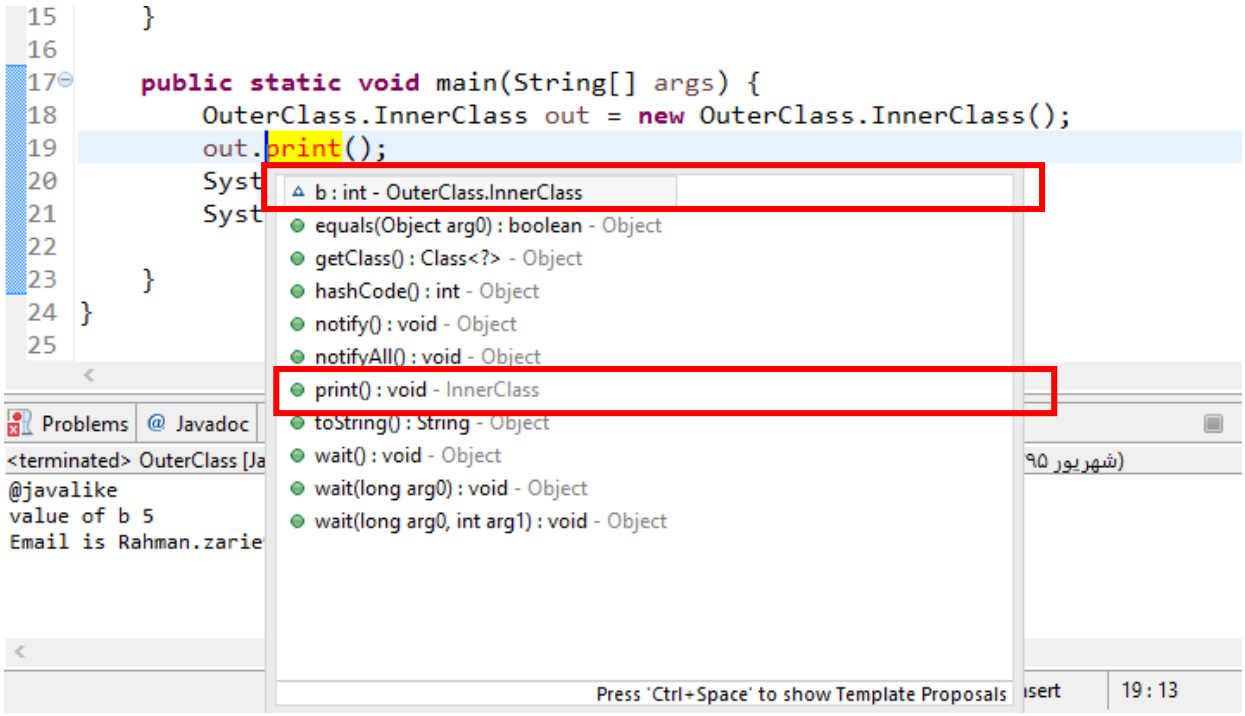
new + نام کلاس بیرونی + . + نام کلاس داخلی استاتیک + ()

```
new OuterClass.InnerClass();
```

در پایان شکل پیاده ساز بصورت زیر درمیاد:

```
OuterClass.InnerClass out = new OuterClass.InnerClass();
```

حال شی out ساخته شده تنها به اعضای کلاس داخلی استاتیک InnerClass دسرسی دارد. تصویر(۱)



تصویر(۱)

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 ازکانال

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.