

به نام خدا

آموزش دوازدهم

در قسمت پیشین دیدیم که به چه نحوی می توان یک شیء از روی کلاس Scanner ساخت و آن را در برنامه خود مورد استفاده قرار داد. در این قسمت قصد داریم به مباحث تکمیلی این کلاس بپردازیم. خیلی اوقات در برنامه نویسی پیش می آید که ما از کاربر انتظار داریم تا فقط داده ای که از وی خواسته شده است را وارد برنامه کند. به عبارت دیگر زمانی که از کاربر می خواهیم که در بخشی مثلاً کد ملی خود را وارد کند، دیگر انتظار نداریم تا او نام خود را در آن بخش مد نظر وارد کند. در زبان برنامه نویسی جاوا این کار را می توان با تعریف نوع ورودی در کلاس Scanner انجام داد.

برای شروع کار پروژه ای تحت عنوان 12th Session به معنی **جلسه دوازدهم** ایجاد کرده و کلاسی تحت عنوان ScannerClass در آن می سازیم. در ابتدا کد ما می بایست به شکل زیر باشد:

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

اکنون همانطور که در جلسات گذشته آموزش داده شد یک شیء از روی کلاس Scanner ساخته و نام آن را keyboardInput به معنی **ورودی کیبورد** می گذاریم. حال کد تکمیل شده ما می بایست به صورت زیر باشد:

```
import java.util.Scanner;  
  
class ScannerClass {  
    public static void main(String[] args) {  
        Scanner keyboardInput = new Scanner(System.in);  
    }  
}
```

دوره آموزش جاوا

کلیه حقوق متعلق به وب سایت نردبان است.

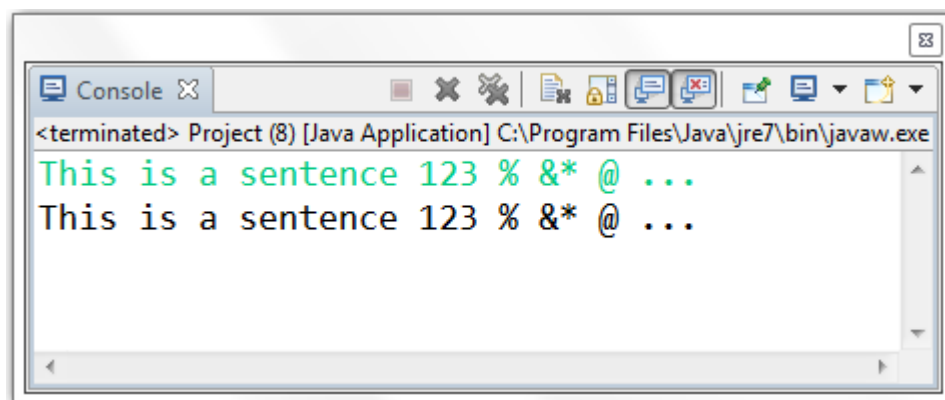
مدرس: بهزاد مرادی

در این قسمت از آموزش قصد داریم تا انواع متدهایی را که می توان به همراه کلاس Scanner مورد استفاده قرار داد را مورد بررسی قرار دهیم. متدی که در جلسه گذشته مورد استفاده قرار دادیم عبارت بود از `nextLine()` به طوریکه یک خط از نوشته را برای ما در پنجره Console به نمایش در آورد. به عبارت دیگر این متد تا زمانی که ما دکمه Enter را نزنیم، هر داده ای را در خود ذخیره خواهد ساخت. به منظور به کار گیری این متد کد خود را به شکل زیر تکمیل می کنیم:

```
import java.util.Scanner;

class ScannerClass {
    public static void main(String[] args) {
        Scanner keyboardInput = new Scanner(System.in);
        System.out.println(keyboardInput.nextLine());
    }
}
```

در کد فوق در دستور `System.out.println()` که وظیفه دارد هر آنچه داخل آن قرار گرفت را روی صفحه مانیتور به نمایش در آورد، نام شیئی خود که از روی کلاس Scanner ساختیم را وارد کرده یک نقطه قرار می دهیم و متد `nextLine()` را به آن ضمیمه می کنیم. در این متد حتی اگر ترکیبی از متن و عدد هم وارد کنیم نمایش داده خواهد شد. به تصویر زیر توجه کنید:



همانطور که در تصویر فوق مشاهده می شود با استفاده از این متد می توان ترکیبی از حروف، کلمات، اعداد و حتی علائم را وارد برنامه خود کرد. این متد زمان هایی که از کاربر بخواهیم مثلاً یک رمز عبور برای خودش تعیین کند بسیار کاربرد خواهد داشت. به طور خلاصه این متد

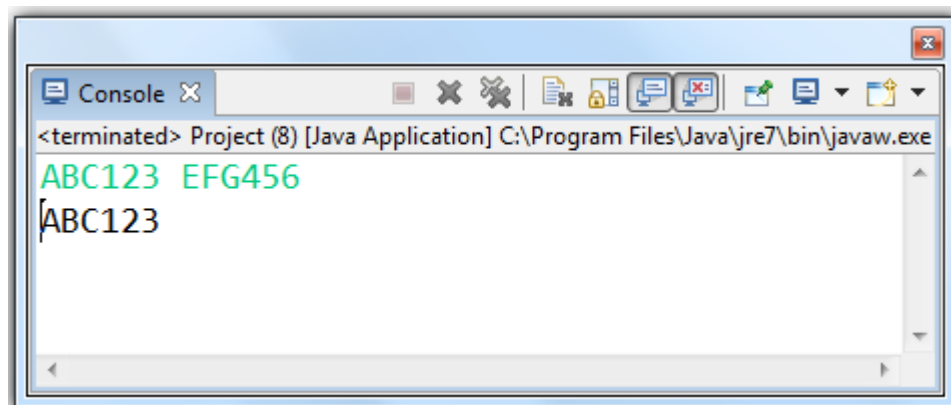
هرآنچه را که کاربر از طریق کیبورد وارد برنامه کند را تا زمانی که کاربر دکمه Enter را بزند دریافت می کند.

متد بعدی متد next() است. از این متد هم می توان برای گرفتن هر نوع ورودی استفاده کرد با این تفاوت که این متد گرفتن داده ها را از کیبورد تا زمانی ادامه می دهد که یک Space مشاهده کند و به محض اینکه یک Space مشاهده شد، گرفتن داده های ورودی متوقف خواهد شد. برای استفاده از این متد کد خود را به شکل زیر بازنویسی می کنیم:

```
import java.util.Scanner;

class ScannerClass {
    public static void main(String[] args) {
        Scanner keyboardInput = new Scanner(System.in);
        System.out.println(keyboardInput.next());
    }
}
```

حال برنامه را اجرا کرده و تعدادی کاراکتر و عدد پشت سر هم نوشته و یک فاصله قرار می دهیم و سپس تعدادی کاراکتر و عدد دیگر می نویسیم سپس دکمه Enter را می زنیم:



همانطور که در تصویر فوق مشاهده می شود، در خط اول عبارت ABC123 وارد شده و سپس عبارت EFG456 و در آخر دکمه Enter را زده ایم. چیزی که متد next() از ورودی برنامه گرفته است به این شکل است که از ابتدای خط شروع به خواندن ورودی کرده تا جاییکه به یک فاصله رسیده است و بلافاصله پس از رسیدن به یک فاصله فرایند گرفتن ورودی متوقف شده است (به عبارت دیگر فاصله ها در این متد نقش Separator یا جدا کننده را بازی می کنند). از اینجا به بعد اگر ما هزاران کاراکتر هم وارد برنامه خود کنیم در برنامه ذخیره نخواهد شد. حال

دوره آموزش جاوا

کلیه حقوق متعلق به وب سایت نردبان است.

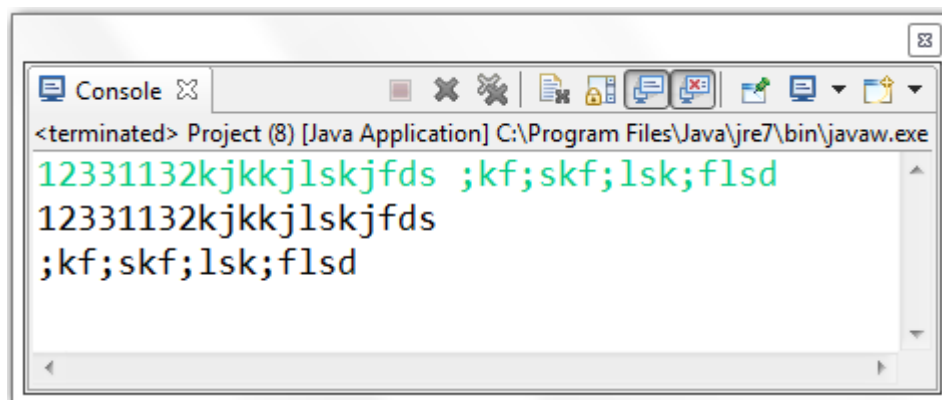
مدرس: بهزاد مرادی

پس از زدن دکمه Enter برنامه فقط داده اول را نمایش می دهد. اکنون اگر بخواهیم کاری کنیم که برنامه ما قادر گردد ورودی هایی که پس از یک فاصله نیز وارد می شوند را دریافت کند، صرفاً نیاز است تا یک بار دیگر شیء ساخته شده از روی کلاس Scanner به همراه یک متد next() دیگر ایجاد کنیم. به کد تکمیل شده زیر توجه کنید:

```
import java.util.Scanner;

class ScannerClass {
    public static void main(String[] args) {
        Scanner keyboardInput = new Scanner(System.in);
        System.out.println(keyboardInput.next());
        System.out.println(keyboardInput.next());
    }
}
```

در این کد ما دو بار از متد next() استفاده کرده ایم که با اینکار به برنامه خود دستور می دهیم که متد اول ورودی از طریق کیبورد را دریافت کند تا جاییکه یک فاصله مشاهده کند. متد دوم به دنبال یک Space می گردد و به محض اینکه اولین فاصله ورودی را پیدا کرد حال هر آنچه پس از آن فاصله وارد شده باشد را گرفته و ذخیره می سازد تا جاییکه یک فاصله دیگر مشاهده کند. به خروجی کد فوق توجه کنید:



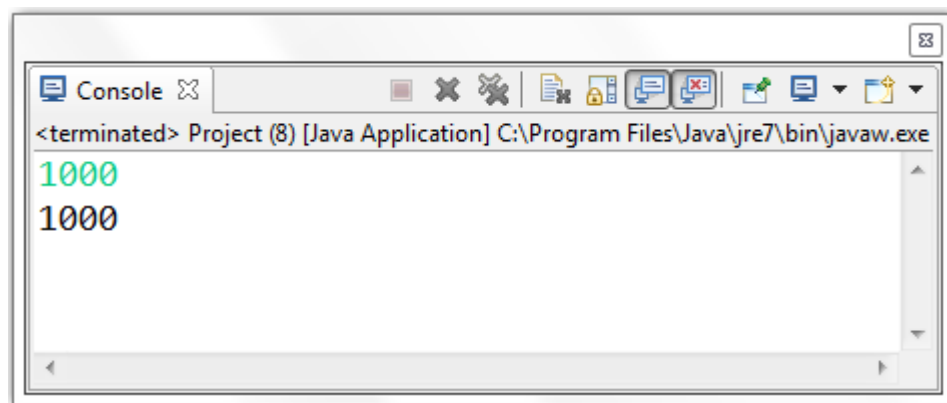
در اجرای فوق تعدادی کاراکتر و عدد تصادفی وارد کرده سپس یک فاصله قرار داده و تعدادی کاراکتر و عدد تصادفی دیگر وارد می کرده ایم و سپس دکمه Enter را زده ایم. کاراکترها و اعداد تصادفی اول تا قبل از فاصله توسط متد next() اول ذخیره شده اند و کاراکترها و اعداد

تصادفی پس از فاصله توسط متد `next()` دوم ذخیره شده اند. پس از زدن دکمه `Enter` ورودی متد اول در خط دوم نمایش داده می شود و ورودی متد دوم در خط سوم نمایش داده می شود. به همین روال می تواند به تعداد نامحدود از متد `next()` استفاده کرد. متد دیگری که می توان مورد استفاده قرار داد متد `nextInt()` است که صرفاً از طریق آن می توان اعداد صحیح را از طریق کیبورد وارد برنامه خود کرد. برای روشن شدن این مطلب کد خود را به شکل زیر تغییر می دهیم:

```
import java.util.Scanner;

class ScannerClass {
    public static void main(String[] args) {
        Scanner keyboardInput = new Scanner(System.in);
        System.out.println(keyboardInput.nextInt());
    }
}
```

اکنون می توانیم پس از اجرای برنامه یک عدد از جنس عدد صحیح وارد برنامه کنیم و سپس دکمه `Enter` را زده و خروجی برنامه را مشاهده کنیم. به طور مثال ما عدد ۱۰۰۰ را وارد می کنیم و دکمه `Enter` را می زنیم:

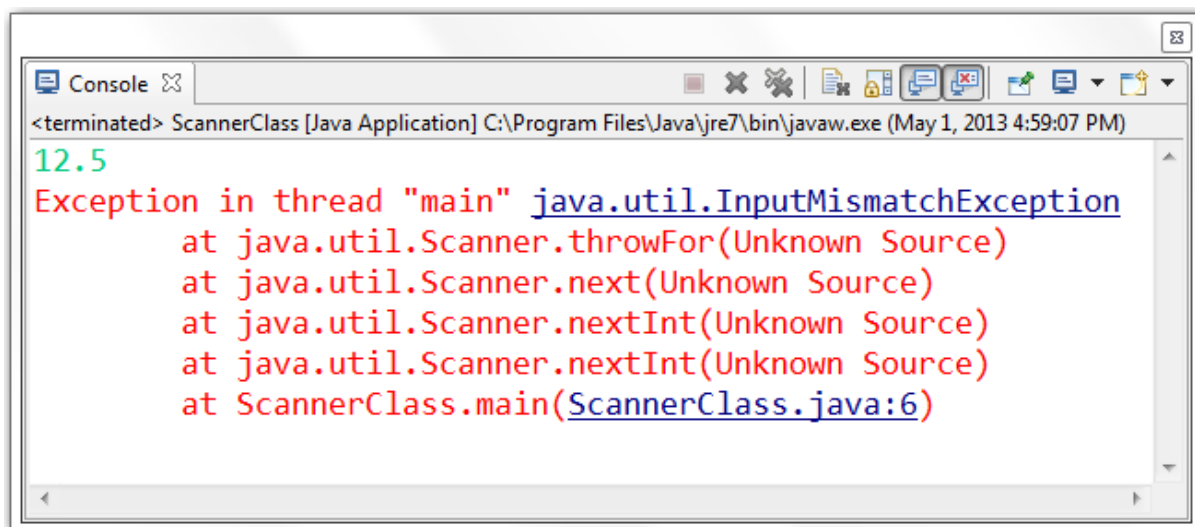


همانطور که گفتیم داده ورودی از جنس عدد صحیح گرفته شده و مجدداً روی صفحه مانیتور نمایش داده می شود. هم اکنون می خواهیم ببینیم که اگر داده ای به غیر از عدد صحیح وارد برنامه کنیم چه اتفاقی خواهد افتاد. برای همین منظور یک بار دیگر برنامه را اجرا کرده و این بار به جای یک عدد صحیح داده ی دیگری مثلاً یک عدد اعشاری وارد مثل عدد 12.5 را وارد می کنیم:

دوره آموزش جاوا

کلیه حقوق متعلق به وب سایت نردبان است.

مدرس: بهزاد مرادی



می بینیم که برنامه ما با یک خطا مواجه شد. در واقع عبارت آبی رنگ
java.util.InputMismatchException در تصویر فوق حاکی از آن است که **داده**
ورودی با متد به کار گرفته شده هم خوانی ندارد. اکنون می بینیم که این نکته چه قدر
اهمیت دارد که نوع داده ورودی با متد استفاده شده در کلاس Scanner می بایست اصطلاحاً با
یکدیگر Match باشند.

متدی دیگری که آشنایی با آن اهمیت دارد متد nextDouble() است به طوریکه این متد
مسئول ذخیره سازی اعداد اعشاری می باشد. برای امتحان کردن این متد کل خود را به شکل زیر
بازنویسی می کنیم:

```
import java.util.Scanner;

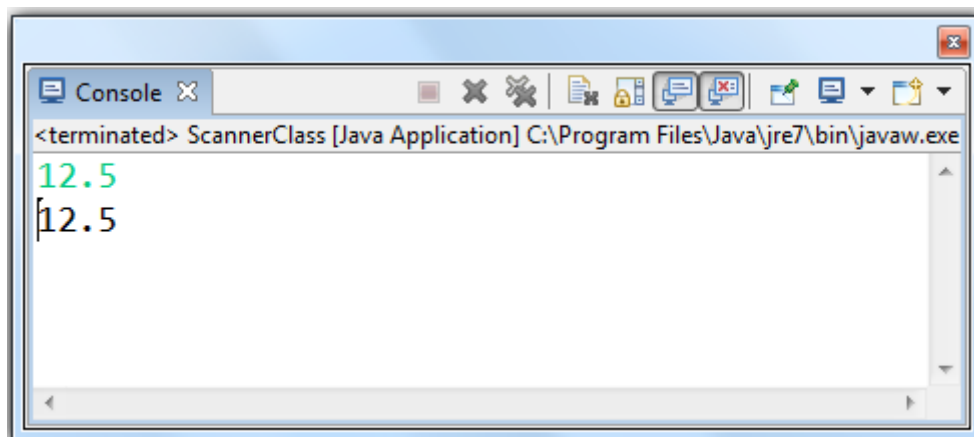
class ScannerClass {
    public static void main(String[] args) {
        Scanner keyboardInput = new Scanner(System.in);
        System.out.println(keyboardInput.nextDouble());
    }
}
```

اکنون پس از اجرای برنامه و وارد کردن یک عدد اعشاری مثل 12.5 خواهیم دید که متد
nextDouble() ورودی اعشاری را در خود ذخیره کرده و سپس از طریق کلاس Scanner
آن را نمایش می دهد:

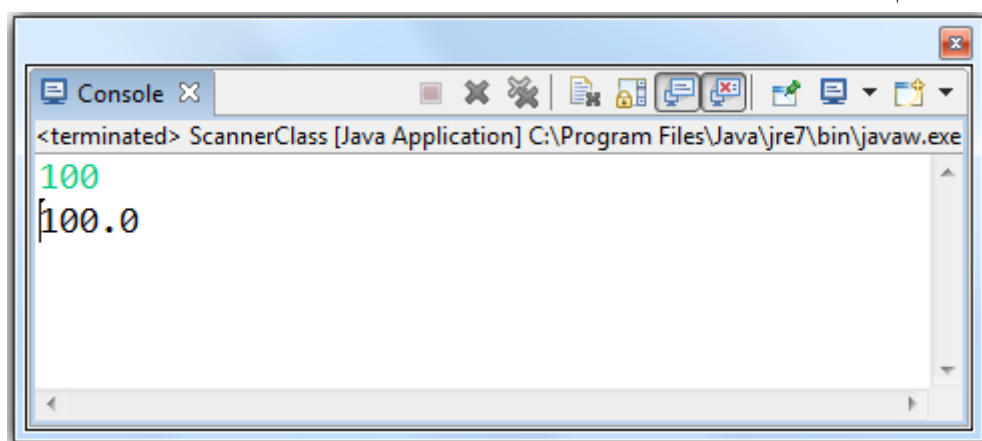
دوره آموزش جاوا

کلیه حقوق متعلق به وب سایت نردبان است.

مدرس: بهزاد مرادی



اگر به خاطر داشته باشید زمانیکه در متد `nextInt()` یک عدد اعشاری وارد کردیم، برنامه از ما خطا گرفت چرا که نوع داده ورودی با نوع متد هم خوانی نداشت. حال ببینیم اگر در متد `nextDouble()` یک عدد صحیح وارد کنیم چه اتفاقی خواهد افتاد (برای مثال عدد ۱۰۰ را وارد می کنیم):



می بینیم که برنامه به درستی کار کرده و حتی عدد صحیح ما را هم به یک عدد اعشاری تبدیل کرده است که مقدار آن عدد اعشاری همانند عدد صحیحی بود که از طریق کیبورد وارد برنامه کردیم با این تفاوت که یک ممیز پس از عدد ورودی قرار داده و یک صفر پس از ممیز گذاشته شده است.

در واقع ما به عنوان یک برنامه نویس حرفه ای این اجازه را نداریم تا برنامه ای طراحی کنیم که در صورت ورود داده اشتباه برنامه Crash کرده و از ادامه باز ایستد. به عبارت دیگر ما موظف هستیم از متدهایی استفاده کنیم که هم خوانی داده ورودی با نوع متغیر را سنجیده و در صورت عدم تطابق به کاربر اخطار دهد. از آنجا که این مبحث نیازمند آشنایی با دستورات `if` نیز می باشد،

مبحث مربوط به متدهای boolean مرتبط با کلاس Scanner پس از آشنایی با دستورات if در قالب آموزش هفدهم بیان خواهد شد. در قسمت آینده به منظور درک بهتر مفاهیم تئوری، پروژه ای تعریف کرده و کلاس Scanner را در آن مورد استفاده قرار می دهیم.