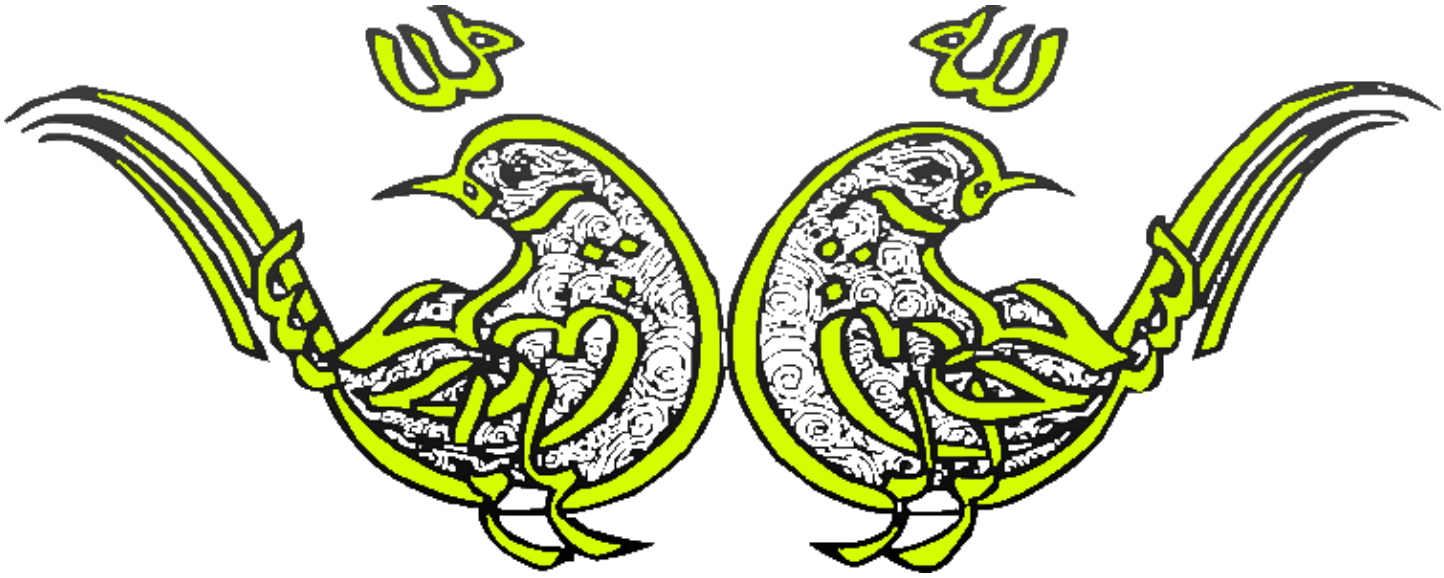


تقدیم به همه هموطنان عزیزم

آموزش زبان برنامه نویسی جاوا

گرافیک در جاوا – پکیج Swing

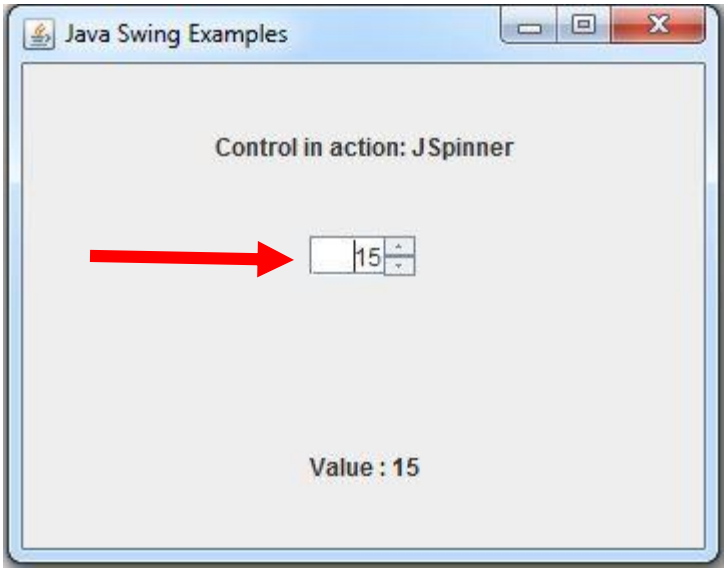
جلسه نوزدهم

کلاس JSpinner

نویسنده : رحمان زارعی

جاوا را ساده،آسان و شیرین بنوشید!!!!

کلاس JSpinner اجزای (component) گرافیکی است که به کاربر اجازه می دهد یک عدد یا یک مقدار را از یک دنباله مرتب از پیش تعیین شده انتخاب کند. برای درک بهتر از ظاهر JSpinner تصویر(۱) را مشاهده کنید:



تصویر(۱)

• سازنده های پرکاربرد کلاس JSpinner :

کاربرد	سازنده
ایجاد یک spinner با یک مدل عدد Integer (صحیح) همراه با مقدار اولیه 0 و بدون محدودیت minimum یا maximum	JSpinner()
ایجاد یک spinner با یک مدل داده ای مشخص	JSpinner(SpinnerModel model)

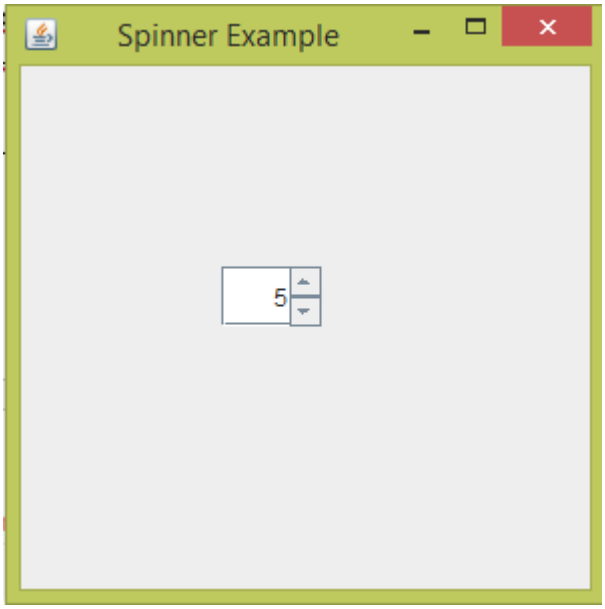
• متدهای پرکاربرد کلاس JSpinner :

کاربرد	متد
اضاف کردن یک Listener به لیستی از Spinner که با هربار تغییر مدل ، عمل مشخصی رخ می دهد.	void addChangeListener(ChangeListener listener)
مقدار فعلی یک Spinner را برمیگرداند.	Object getValue()

مثال:

```
package javalike;
import javax.swing.*;
public class SpinnerExample {
    public static void main(String[] args) {
    JFrame f=new JFrame("Spinner Example");
    SpinnerModel value =
             new SpinnerNumberModel(5, //initial value
                0, //minimum value
                10, //maximum value
                1); //step
    JSpinner spinner = new JSpinner(value);
            spinner.setBounds(100,100,50,30);
            f.add(spinner);
            f.setSize(300,300);
            f.setLayout(null);
            f.setVisible(true);
}
}
```

خروجی:تصویر(۲)



تصویر(۲)

توضیحات:

```
SpinnerModel value = new SpinnerNumberModel(5, 0, 10, 1);
```

- برای ایجاد یک مدل داده ای برای Spinner خود از کلاس SpinnerModel شی ایجاد می کنیم.
- پارامتر سازنده کلاس SpinnerModel به ترتیب عبارت است از :

۱. عدد ۵ برای مقداردهی اولیه به Spinner می باشد. هنگام نمایش Spinner مقدار انتخاب شده پیشفرض آن ۵ می باشد.

۲. عدد 0 minimum مقدار Spinner ما می باشد. یعنی مقدار Spinner را نمی توانیم کمتر از عدد صفر تغییر دهیم.

۳. عدد ۱۰ maximum مقدار Spinner ما می باشد. یعنی مقدار Spinner را نمی توانیم بیشتر از عدد ده تغییر دهیم.

۴. با توجه به موارد ۲ و ۳ ، محدوده مقادیر Spinner ما بین ۰ تا ۱۰ می باشد و تنها در این بازه می توانیم مقادیر Spinner را تغییر دهیم.

۵. عدد ۱ گام های تغییر مقدار Spinner ما را تعیین می کند. مثلا عدد ۱ باعث می شود یکی یکی مقادیر Spinner خود را تغییر دهیم. اگر مثلا عدد ۳ بود سه تا سه تا مقادیر Spinner تغییر میکرد.

```
JSpinner spinner = new JSpinner(value);
```

- از کلاس JSpinner شی ایجاد کرده و شی value که از نوع SpinnerModel می باشد و مدل داده ای Spinner را مشخص می کند به عنوان پارامتر به سازنده کلاس JSpinner داده ایم.

```
1.    spinner.setBounds(100,100,50,30);
2.            f.add(spinner);
3.            f.setSize(300,300);
4.            f.setLayout(null);
5.            f.setVisible(true);
```

۱. تعیین مختصات و ابعاد spinner

۲. اضاف کردن spinner به فریم

۳. تعیین اندازه فریم

۴. چون فعلا از طرح بندی خاصی استفاده نکردیم مقدار پارامتر این متد را null قرار می دهیم.

۵. برای نمایش فریم و تمام اجزای گرافیکی از این متد استفاده می کنیم.

مثال:

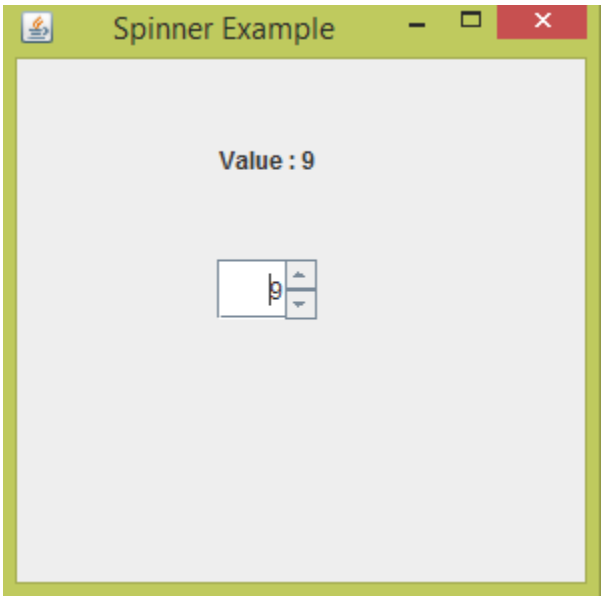
```
package javalike;

import javax.swing.*;
import javax.swing.event.*;

public class SpinnerExample {
      public static void main(String[] args) {
            JFrame f = new JFrame("Spinner Example");
            final JLabel label = new JLabel();
            label.setHorizontalAlignment(JLabel.CENTER);
            label.setSize(250, 100);
            SpinnerModel value = new SpinnerNumberModel(5, // initial value
                        0, // minimum value
                        10, // maximum value
                        1); // step
            JSpinner spinner = new JSpinner(value);
            spinner.setBounds(100, 100, 50, 30);
            f.add(spinner);
```

```
            f.add(label);
            f.setSize(300, 300);
            f.setLayout(null);
            f.setVisible(true);
            spinner.addChangeListener(new ChangeListener() {
                  public void stateChanged(ChangeEvent e) {
                        label.setText("Value : "
                                    + ((JSpinner) e.getSource()).getValue());
                  }
            });
      }
}
```

خروجی: تصویر(۳)



تصویر(۳)

توضیحات:

```
1. label.setHorizontalAlignment(JLabel.CENTER);
2. label.setSize(250, 100);
```

۱. با این دستور `label` ما در وسط مختصات افقی در فریم قرار میگیرد.

۲. تعیین اندازه `label`

```
spinner.addChangeListener(new ChangeListener() {
            public void stateChanged(ChangeEvent e) {
                  label.setText("Value : "
                              + ((JSpinner) e.getSource()).getValue());
            }
      });
```

- متد addChangeListener برای افزودن یک ChangeListener به spinner ما کاربرد دارد.
- ChangeListener یک اینترفیس است که از طریق متد stateChanged که در بدنه آن قرار دارد رویدادهای مربوط به تغییر مقدار spinner را دریافت می کند.
- با هر بار رخ دادن رویدادی روی spinner متد stateChanged صدا زده می شود و دستورات درون آن اجرا می شود.

```
label.setText("Value : "+ ((JSpinner) e.getSource()).getValue());
```

- این دستور مقدار انتخاب شده spinner را درون label نمایش می دهد.
- خب چطور این کار انجام می شود؟!

وقتی رویدادی رخ می دهد درون شی e که از نوع کلاس ChangeEvent هستش ریخته می شود.حال ما از طریق شی e متد getSource را صدا زده ایم ، این متد همون طور که از اسمش پیداست میگرده منبع رویداد (منظور همون اجزای گرافیکی که رویدادی روی آن آنها اتفاق افتاده است) را پیدا میکند و بصورت یک شی از نوع کلاس Object (پدر همه کلاس هاست) برمیگرداند. یک مشکل کوچک پیش اومده! مقدار برگردانده شده از نوع کلاس Object می باشد و از طرفی اجزای گرافیکی ما از نوع کلاس JSpinner هستش! چکار باید کنیم؟! باید شی از نوع کلاس Object را به شی از نوع کلاس JSpinner تبدیل می کنیم. چگونه؟ با استفاده از روش casting یا تبدیل نوع، به سادگی نوع کلاس خود را درون پرانتز در کنار شی کلاسی که قصد داریم آن را تبدیل کنیم قرار می دهیم. مثلا در اینجا کلاس JSpinner درون پرانتز در کنار e.getSource که شی کلاس Object را برمیگرداند قرار داده ایم. حالا بعد از تبدیل شی از نوع کلاس Object به شی از نوع کلاس JSpinner به راحتی می توانیم متد getValue را صدا بزنیم.متد getValue مقدار انتخاب شده Spinner را برمی گرداند.

حال از طریق شی label متد setText را صدا زده و رشته "Value : " همراه با مقدار spinner را درون لیبل نمایش می دهیم.

- اگر توضیحات گیج کننده بود نگران نباشید!! تنها با نگاه کردن به کد و تغییر دادن کد روش کار دستتون میاد. شما میگید مثال رو نگاه کن و دستکاری کن! تا یاد بگیری! مثلا چکار کنم؟! ی نمونه دستکاری که خودم انجام دادم که مفهوم دستور زیر رو یادبگیرم را در مثال بعدی نگاه کنید:

```
((JSpinner) e.getSource()).getValue()
```

مثال:

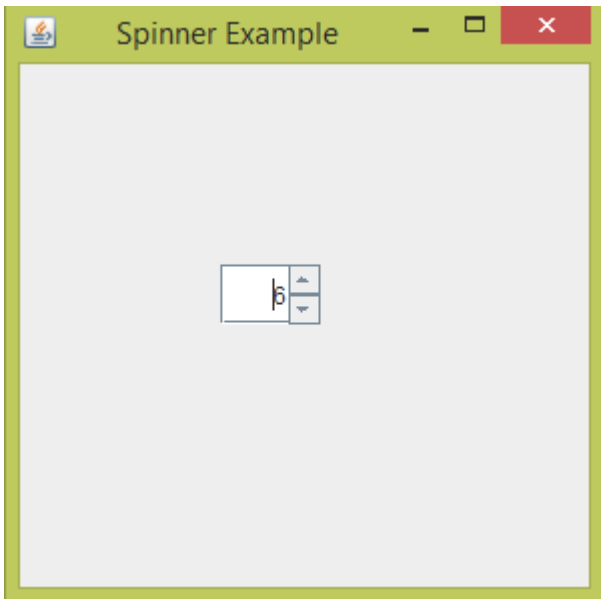
```
package javalike;

import javax.swing.*;
import javax.swing.event.*;

public class SpinnerExample {
      public static void main(String[] args) {
            JFrame f = new JFrame("Spinner Example");
            SpinnerModel value = new SpinnerNumberModel(5, // initial value
                        0, // minimum value
                        10, // maximum value
                        1); // step
            JSpinner spinner = new JSpinner(value);
            spinner.setBounds(100, 100, 50, 30);
            f.add(spinner);
            f.setSize(300, 300);
            f.setLayout(null);
            f.setVisible(true);
            spinner.addChangeListener(new ChangeListener() {
                  public void stateChanged(ChangeEvent e) {

                        System.out.println(e.getSource());
                        System.out.println((JSpinner) e.getSource());
                        System.out.println(((JSpinner) e.getSource()).getValue());
                  }
            });
      }
}
```

خروجی: بعد از اجرای برنامه اگر مقدار spinner را به عدد ۶ تغییر دهیم خروجی به صورت تصویر(۴) خواهد بود:



تصویر(۴)

- مقدار پیشفرض این spinner ابتدا ۵ بود که به عدد ۶ تغییر دادیم که نتیجه این تغییر در محیط کنسول به صورت زیر است:

خروجی در کنسول:

```
javax.swing.JSpinner[,100,100,50x30,invalid,layout=javax.swing.plaf.basic.BasicSpinnerUI$Handler,alignmentX=0.0,alignmentY=0.0,border=javax.swing.plaf.BorderUIResource$CompoundBorderUIResource@737d0985,flags=328,maximumSize=,minimumSize=,preferredSize=]

javax.swing.JSpinner[,100,100,50x30,invalid,layout=javax.swing.plaf.basic.BasicSpinnerUI$Handler,alignmentX=0.0,alignmentY=0.0,border=javax.swing.plaf.BorderUIResource$CompoundBorderUIResource@737d0985,flags=328,maximumSize=,minimumSize=,preferredSize=]

6
```

- خروجی سبز رنگ نتیجه دستور زیر می باشد، خروجی سبز آدرس شی است که متد getSource برگردانده است.

```
System.out.println(e.getSource());
```

- خروجی آبی رنگ نتیجه دستور زیر می باشد، خروجی آبی آدرس شی بعد از عمل casting است.

```
System.out.println(((JSpinner) e.getSource()));
```

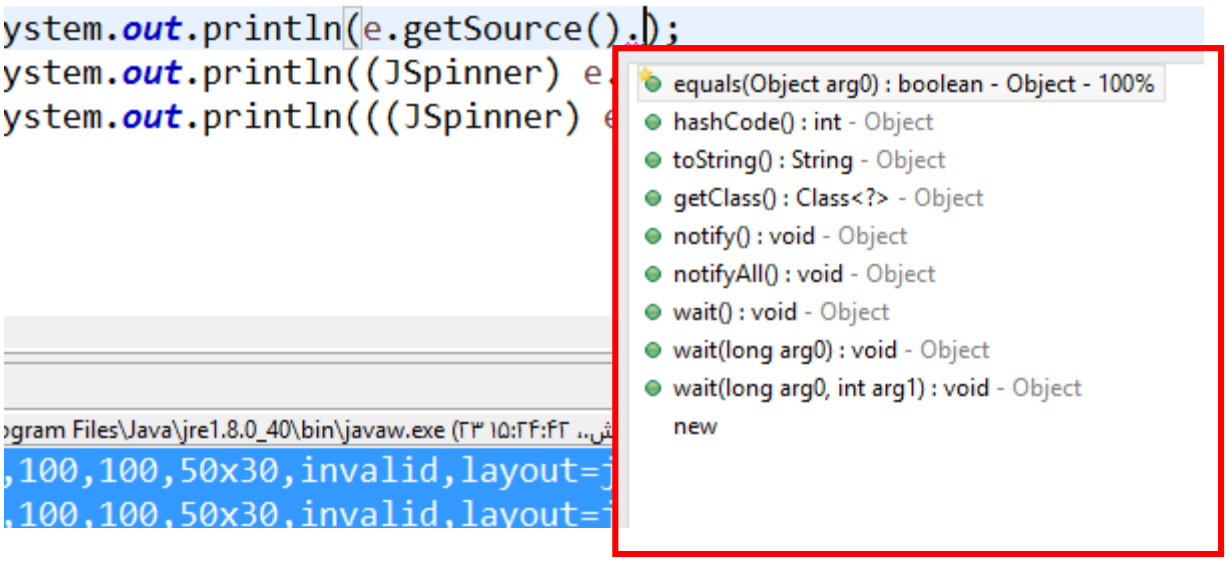
- خروجی صورتی رنگ نتیجه دستور زیر می باشد:

```
System.out.println(((JSpinner) e.getSource()).getValue());
```

توضیحات:

```
System.out.println(e.getSource());
```

- این دستور منبعی که رویدادی روی آن رخ داده است را برمیگرداند و چاپ می کند.
- متد **getSource** منبع رویداد رخ داده را برمی گرداند.
- این منبع میتونه یک اجزای گرافیکی نظیر **JSpinner** و... باشد.
- همان طور که در جلسات گذشته گفتیم کلاس **Object** پدر همه کلاس ها می باشد. به عبارتی همه کلاس های درون جاوا زیر کلاس و فرزند کلاس **Object** می باشد.
- خب **getSource** شی که برمیگرداند از نوع کلاس **Object** می باشد. این شی میتونه یکی از اجزای گرافیکی در جاوا باشد ولی نمی توانیم مستقیم به متدها و ویژگی های آن اجزای گرافیکی دسرسی داشته باشیم و باید حتما عمل **casting** را انجام دهیم. برای درک سطح دسرسی شی که این متد بدون عمل **casting** برای ما برمیگرداند تصویر(۵) را مشاهده کنید:

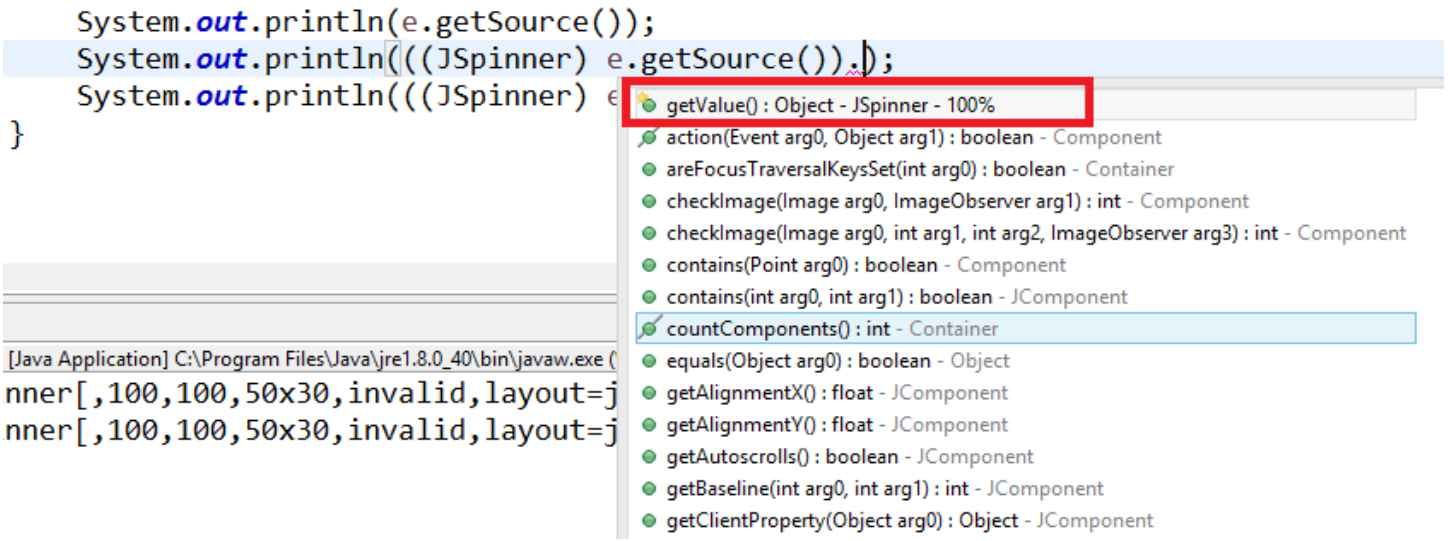


تصویر(۵)

همان طور که در تصویر(۵) مشاهده می کنید شی که این متد برمیگرداند فقط به ویژگیها و رفتارهای کلاس Object دسرسی دارد و خبری از متدها و ویژگیهای اجزای گرافیکی خاصی نیست! حال عمل casting را به صورت زیر انجام میدهیم:

```
System.out.println(((JSpinner) e.getSource()));
```

- این دستور منبع رویدادی که متد getSource برمی گرداند را به شی از کلاس JSpinner تبدیل می کند.
- حالا سطح دسرسی منبع دریافت شده و تبدیل آن به JSpinner را در تصویر(۶) مشاهده کنید:



تصویر(۶)

- همان طور که در تصویر(۶) مشاهده می کنید با عمل casting به انبوهی از متدهای اجزای گرافیکی از جمله به متد getValue دسرسی داریم.
- جالب است بدانید که خروجی دو دستور زیر شبیه به هم هستند اما در عمل متفاوت! :

```
System.out.println(e.getSource());
System.out.println(((JSpinner) e.getSource()));
```

- حال با صدا زدن متد getValue مقدار spinner را دریافت می کنیم:

```
System.out.println(((JSpinner) e.getSource()).getValue());
```

خب چطور بود؟! پیشنهاد میکنم هنوز مثال های بیشتری ببینید کسی با یکی دوتا مثال برنامه نویس نشده!

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.

دخل و تصرف ، ویرایش و کپی زدن تمامی آموزش های جاوالایک به دور از اخلاق حرفه ای ست و حرام می باشد.