# آموزش زبان برنامه نویسی جاوا

# گرافیک در جاوا – پکیج Swing

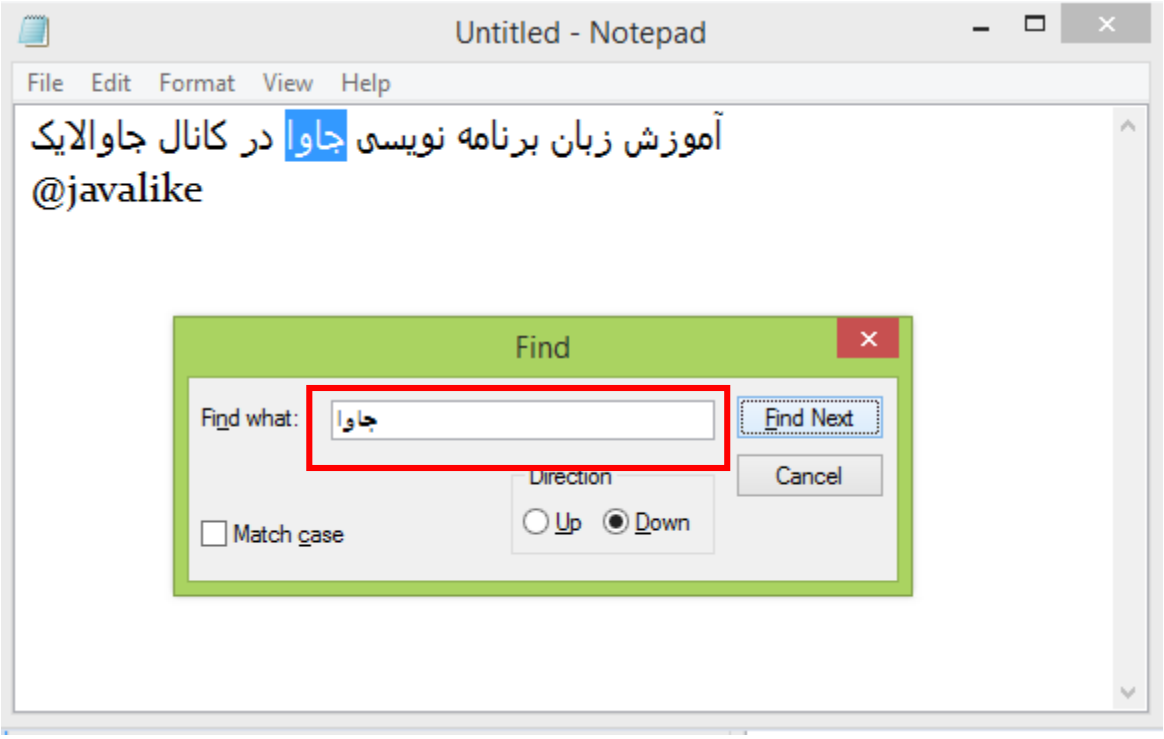
## جلسه چهارم

## کلاس JTextField

نویسنده : رحمان زارعی

جاوا را ساده،آسان و شیرین بنوشید!!!!

کلاس JTextField اجزای گرافیکی "متنی" است که با شی ساختن از آن به شما اجازه می دهد یک خط متن را ویرایش کنید.برای تصوری بهتر از این اجزای گرافیکی برنامه Notepad را باز می کنیم و از منو Edit گزینه Find را انتخاب می کنیم:تصویر(۱)




تصویر(۱)

در تصویر(۱) ما قصد داریم کلمه "جاوا" را در میان متن موجود در برنامه Notepad پیدا کنیم.فیلدی که در بخش find ، کلمه مورد نظر را برای پیدا شدن وارد می کنیم همان cpmponent یا اجزای گرافیکی JTextField هستش که با رنگ قرمز مشخص کرده ایم. احتمالا با همچین اجزای گرافیکی زیاد برخورد کرده باشید مثلا هنگام ثبت نام و وارد کردن اطلاعات کاربری . متنی که در این اجزای گرافیکی وارد میکنیم دارای ویژگی های زیر می باشد:

۱. بصورت یک خط یا یک سطر می باشد.
۲. تک متن هستش یعنی نقطه سر خط نداریم! ☺ سطر بعد ندارد! یعنی نمی توانیم متنی وارد آن کنیم و متن جدیدی در سطر بعد وارد کنیم!
۳. بر خلاف label توسط کاربر نیز قابل ویرایش هستش یعنی شما میتوانید متن مورد نظر را جایگزین متن قبل کنید.

- خب حالا که ذهنیتی نسبت کلاس JTextField پیدا کردیم سراغ سازنده ها ، متدها و روش پیاده سازی آن در جاوا می رویم.

## سازنده های پرکاربرد کلاس JTextField :

| سازنده | JTextField() |
|---|---|
| توصیف | برای ایجاد یک TextField جدید بدون پارامتر ( سفید یا همون خالی ) |

| سازنده | JTextField(String text) |
|---|---|
| توصیف | برای ایجاد یک TextField جدید با یک متن خاص |

| سازنده | JTextField(String text, int columns) |
|---|---|
| توصیف | برای ایجاد یک TextField جدید با یک متن و ستون خاص |

| سازنده | JTextField(int columns) |
|---|---|
| توصیف | برای ایجاد یک TextField جدید با یک ستون خاص |

## متد های پرکاربرد کلاس JTextField :

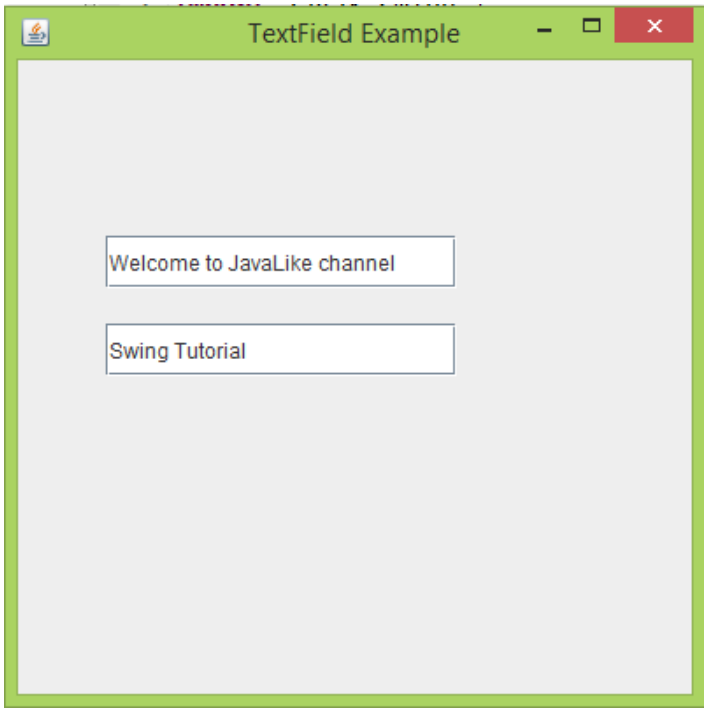
| متد | void setFont(Font f) |
|---|---|
| توصیف | برای تنظیم فونت TextField استفاده می شود. |

| متد | void addActionListener(ActionListener l) |
|---|---|
| توصیف | این متد نیز کاربردش برای TextField تقریبا شبیه Button هستش.یعنی اجرای دستورات و عملیات خاص باتوجه به رویدادی که اتفاق می افتاد.مثلا در با Button با کلیک کردن روی دکمه مورد نظر دستور خاص مربوط به ان دکمه اجرا میشد. البته در TextField وقتی شما متن مورد نظر را وارد میکنید و بعد دکه Enter کیبورد را فشار می دهید یک رویداد رخ میدهد و دستورات و عملیات مربوط به TextField اجرا می شود.<br>**در کل اگر میخواید بعد از پر کردن متن TextField مربوطه ، دستور خاصی اجرا شود از این متد استفاده می کنیم.**<br>میدونم احتمالا گیج شدید نگران نباشد با مثال متوجه خواهیم شد. |

مثال :

```
package swing_javalike;
import javax.swing.*;

class TextFieldExample
{
public static void main(String args[])
    {
    JFrame f= new JFrame("TextField Example");
    JTextField t1,t2;
    t1=new JTextField("Welcome to JavaLike channel");
    t1.setBounds(50,100, 200,30);
    t2=new JTextField("Swing Tutorial");
    t2.setBounds(50,150, 200,30);
    f.add(t1); f.add(t2);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
    }
    }
```

خروجی: تصویر(۲)




تصویر(۲)

- همان طور که مشاهده می کنید ما در این برنامه تو TextField داریم که متن دلخواه ما را نمایش می دهند.اگر برنامه رو اجرا کنید خواهید دید که می توانید متن های درون هر دو TextField را ویرایش کنید.
- هر چقدر متن به این دو TextField اضافه کنید به سطر بعد نمی توانیم برویم.

```
JFrame f= new JFrame("TextField Example");
```

- همان طور که میدانید برای ایجاد هر برنامه گرافیکی ابتدا اسکلت برنامه یعنی frame ان را ایجاد می کنیم.متن درون پارامتر سازنده ، عنوان فریم ما را تشکیل می دهد.

```
JTextField t1,t2;
```

- ایجاد دو شی از نوع کلاس JTextField
- این دو شی null هستند و باید با صدا زدن سازنده کلاس آن ها را ایجاد کنیم.

```
t1=new JTextField("Welcome to JavaLike channel");
```

- صدا زدن سازنده کلاس JTextField و ایجاد شی با نام t1
- پارامتر سازنده کلاس متن موجود در فیلد ما را تشکیل می دهد.

```
t1.setBounds(50,100, 200,30);
```

- با استفاده از متد setBounds مختصات و ابعاد اجزای گرافیکی TextField را تنظیم می کنیم.
- در این متد x=50 ، y=100 ، width=200 و height=30

```
t2=new JTextField("Swing Tutorial");
t2.setBounds(50,150, 200,30);
```

- تمام کارایی که برای فیلد t1 انجام دادیم برای فیلد t2 نیز انجام می دهیم.

```
f.add(t1); f.add(t2);
f.setSize(400,400);
f.setLayout(null);
f.setVisible(true);
```

- افزودن component ها (اجزای گرافیکی) TextField های t1 و t2 به frame خود، این کار چیزی شبیه اسکلت خانه است که بهش سایر متریال را اضافه می کنیم.
- تنظیم سایز فریم خود
- چون فعلا قصد نداریم از طرح بندی خاصی استفاده کنیم پارامتر متد setLayout را null قرار داده ایم.

- برای نمایش فریم و تمام اجزای گرافیکی باید متد **setVisible** را با شی ای که از کلاس **JFrame** ساخته ایم صدا زده و مقدار پارامتر آن را **true** قرار دهیم.

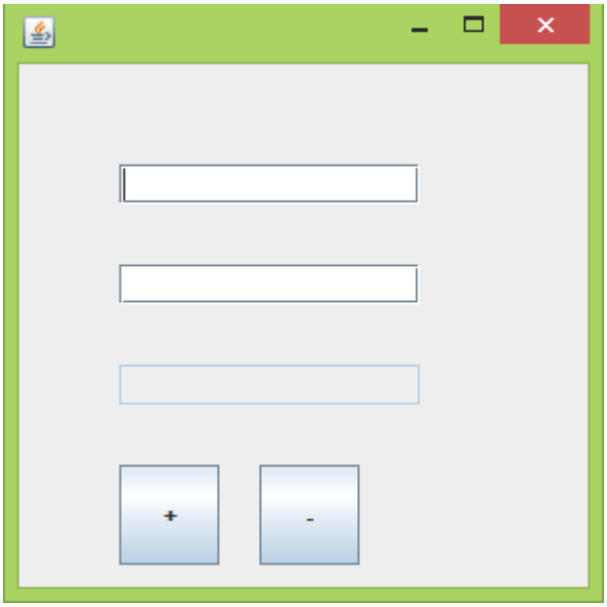
تا اینجا چطور بود؟ ☺ در ادامه با ما همراه شید.

مثال:در برنامه زیر سه **TextField** و دو **Button** داریم که یکی مقادیر دو فیلد اول را با هم جمع و دیگر از هم کم می کند و نتیجه را در فیلد سوم نمایش می دهد.

```
Package swing_javalike;

import javax.swing.*;
import java.awt.event.*;

public class TextFieldEx implements ActionListener {
      JtextField tf1, tf2, tf3;
      JButton b1, b2;

      TextFieldEx() {
            Jframe f = new Jframe();
            tf1 = new JtextField();
            tf1.setBounds(50, 50, 150, 20);
            tf2 = new JtextField();
            tf2.setBounds(50, 100, 150, 20);
            tf3 = new JtextField();
            tf3.setBounds(50, 150, 150, 20);
            tf3.setEditable(false);
            b1 = new JButton(“+”);
            b1.setBounds(50, 200, 50, 50);
            b2 = new JButton(“-“);
            b2.setBounds(120, 200, 50, 50);
            b1.addActionListener(this);
            b2.addActionListener(this);
            f.add(tf1);
            f.add(tf2);
            f.add(tf3);
            f.add(b1);
            f.add(b2);
            f.setSize(300, 300);
            f.setLayout(null);
            f.setVisible(true);
      }

      public void actionPerformed(ActionEvent e) {
            String s1 = tf1.getText();
```

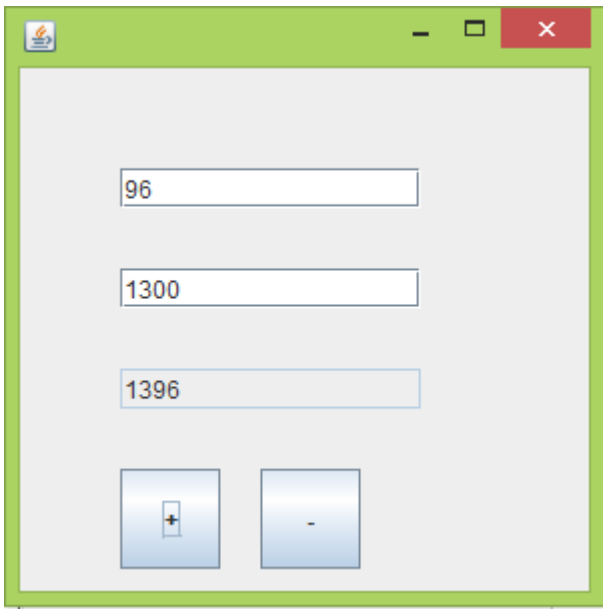
```
            String s2 = tf2.getText();
            int a = Integer.parseInt(s1);
            int b = Integer.parseInt(s2);
            int c = 0;
            if (e.getSource() == b1) {
                  c = a + b;
            } else if (e.getSource() == b2) {
                  c = a - b;
            }
            String result = String.valueOf(c);
            tf3.setText(result);
      }

      public static void main(String[] args) {
            new TextFieldEx();
      }
}
```

خروجی: بعد از اجرای برنامه خروجی بصورت زیر خواهد بود: تصویر(۳)




تصویر(۳)

- برای مثال اگر مقدار فیلد اول را ۹۶ بگذاریم و مقدار فیلد دوم را ۱۳۰۰ بگذاریم و روی دکمه **"+"** کلیک کنی نتیجه در فیلد سوم ۱۳۹۶ خواهد شد: تصویر(۴)




تصویر(۴)

- همان طور که در تصویر(۴) مشاهده می کنید TextField سوم با TextField های اول و دوم شکل متفاوتی دارد و دورش خطوط ابی رنگ و غیر قابل ویرایش می باشد.دلیلش اینه که درهنگام کدنویسی با صدا زدن دستور مربوطه که جلوتر بهش می پردازیم گفتیم که فیلد سوم برای کاربر غیرقابل ویرایش کن!

نکته:حتما مثال های کار با گرافیک را برای خودتون در Eclipse و.. کامپایل و اجرا کنید و عملی کار کنید، چون که تئوری کار ساز نیست.

```
public class TextFieldEx implements ActionListener {
```

- چون که قصد داریم از رویداد ها و اکشن ها استفاده کنیم و عملیات خاصی را به اجزای گرافیکی خود نسبت دهیم کلاس خود را implements به اینترفیس ActionListener می کنیم.

```
JtextField tf1, tf2, tf3;
JButton b1, b2;
```

- در اینجا سه شی از نوع کلاس JTextField تعریف کرده ایم.
- و همچنین دو شی از کلاس JButton که دکمه های ما را تشکیل می دهد تعریف کرده ایم.

```
TextFieldEx() {
            Jframe f = new Jframe();
            tf1 = new JtextField();
            tf1.setBounds(50, 50, 150, 20);
            tf2 = new JtextField();
            tf2.setBounds(50, 100, 150, 20);
            tf3 = new JtextField();
            tf3.setBounds(50, 150, 150, 20);
            tf3.setEditable(false);
            b1 = new JButton(“+”);
            b1.setBounds(50, 200, 50, 50);
            b2 = new JButton(“-“);
            b2.setBounds(120, 200, 50, 50);
```

- در سازنده کلاس اجزای گرافیکی خود را ساخته ایم.
- ابتدا یک frame ایجاد کرده ، و سپس سازنده کلاس JTextField را برای هر کدام از اشیای t1،t2و t3 صدا زده ایم.و سپس مختصات و ابعاد TextField های خود را مشخص کرده و همچنین سازنده کلاس JButton را برای اشیای b1 و b2 صدا زده ایم.و در نهایت تنظیم مختصات و ابعاد Button های خود

```
            b1.addActionListener(this);
            b2.addActionListener(this);
```

- همان طور که میدانید برای نسبت دادن رویداد و عملیات خاصی به یک Button با استفاده از شی ایجاد شده از کلاس JButton متد addActionListener را صدا می زنیم.
- در جلسات گذشته مستقیم از اینترفیس ActionListener درون متد addActionListener شی ایجاد کرده و دستورات خودمون که قرار بود بعد از رخ دادن یک رویداد ( یک رویداد می تواند کلیک کردن روی دکمه Button ، فشردن دکمه کیبورد یا کلیک کردن با موس باشد) اجرا شوند رو درون متد actionPerformed قرار می دادیم.
- در اینجا خبری از این کار نیست یعنی بجای ایجاد مستقیم شی از اینترفیس ActionListener ، کلاس خود را impelements به اینترفیس ActionListener کردیم و در بدنه کلاس خود متد actionPerformed اینترفیس ActionListener را پیاده سازی کرده و دستوراتی که قصد اجرا شدن بعد از رخ دادن هر رویدادی داریم را درون متد actionPerformed قرار می دهیم.
- از آن جهت که کلاس ما یعنی TextFieldEx به اینترفیس ActionListener ، implements شده است، می توانیم شی ای از کلاس خود یعنی TextFieldEx را جایگزین پارامتر متد

addActionListener کنیم. از طرفی بجای شی سازی از کلاسمون در این متد می توانیم از کلمه کلیدی this استفاده کنیم.

```
f.add(tf1);
f.add(tf2);
f.add(tf3);
f.add(b1);
f.add(b2);
f.setSize(300, 300);
f.setLayout(null);
f.setVisible(true);
```

- در اینجا با استفاده از متد add اجزای گرافیکی خود را به frame اضافه می کنیم.
- همچنین سایز فریم خود را تعیین می کنیم.
- چون از طرح بندی خاصی نمی خوایم استفاده کنیم مقدار پارامتر setLayout را null قرار می دهیم.
- برای نمایش فریم و تمام اجزای گرافیکی مقدار پارامتر متد setVisible را true می کنیم.

```
public void actionPerformed(ActionEvent e) {
```

- این متد رویدادهایی که رخ می دهند را دریافت می کند و بعد متناسب با رویداد رخ داده دستوری که بهش داده ایم را اجرا می کند.
- یک رویداد می تواند دکمه های کیبورد، موس و یا دکمه های درون برنامه مثل فشردن Button ها، آیتم های منوها و...باشد.

```
String s1 = tf1.getText();
String s2 = tf2.getText();
```

- با صدا زدن متد getText از طریق شی های کلاس JTextField متن موجود در فیلد ها دریافت می شوند.
- در اینجا متن هر کدام TextField ها درون متغیری از نوع String ریخته می شود.
- مقدار یک TextField از نوع String می باشد حتی اگر عدد وارد آن کنیم.

```
int a = Integer.parseInt(s1);
int b = Integer.parseInt(s2);
```

- با استفاده از متد Integer.parseInt متغیر از نوع رشته (String) را به متغیر از نوع int تبدیل کرده ایم.

- خب همان طور که می دانید داده های یک **textfield** از نوع رشته هستش، از آنجایی که قصد داریم عملیات ریاضی رو این داده ها انجام دهیم ابتدا باید به عدد صحیح تبدیل شوند.

```
    int c = 0;
        if (e.getSource() == b1) {
            c = a + b;
        } else if (e.getSource() == b2) {
            c = a - b;
        }
        String result = String.valueOf(c);

        tf3.setText(result);
    }

    public static void main(String[] args) {
        new TextFieldEx();
    }
```

- در این کد بالا کارهای زیر انجام می گیرد:

۱. دریافت رویداد مربوط به هر **Button**

۲. تشخیص این که این رویداد مربوط به کدام یک از **button** ها می شود

```
if (e.getSource() == b1) {
            c = a + b;
        } else if (e.getSource() == b2) {
            c = a - b;
        }
```

- شی **e** از نوع **ActionEvent** هستش که هر رویداد بعد از رخ دادن در آن قرار می گیرد.
- با صدا زدن متد **getSource** منبع رویدادی که رخ داده است را تشخیص می دهیم.

۳. اگر رویداد رخ داده مربوط به دکمه **b1** باشد دو عدد دریافت کرده از فیلدها رو با هم جمع می زند.

```
if (e.getSource() == b1) {
c = a + b;
        }
```

۴. در غیر این صورت اگر رویداد رخ داده مربوط به دکمه **b2** باشد دو عدد دریافت کرده از فیلدها رو از هم کم می کند

```
else if (e.getSource() == b2) {
            c = a - b;
        }
```

۵. نتیجه حاصل جمع یا تفریق را به رشته تبدیل کرده و درون متغیری از نوع رشته می ریزیم:

```
String result = String.valueOf(c);
```

۶. نتیجه محاسبات دو عدد را در textfield سوم می ریزیم.

```
tf3.setText(result);
```

۷. در پایان در متد main از کلاسمون شی ایجاد می کنیم و برنامه کاربردی که نوشتیم اجرا می شود.

- دوستان من تنها راه یادگیری گرافیک در جاوا دیدن مثال های فراون و روش های حل آنها می باشد،تئوری جز گیج کردن چیزی دیگری در بر ندارد.
- در مبحث گرافیک شما بصورت کاربردی از شی گرایی جاوا استفاده می کنید. مثلا هنگام ساختن یک دکمه button بجای نوشتن تمامی دستورات کلاس button برای هر یک از دکمه ها ،تنها یک کلاس button نوشته شده است که روی آن به تعداد دلخواه شی ایجاد می کنیم و در برنامه مون استفاده می کنیم.پس شی گرایی اومده که کد زدن رو برامون ساده و کوتاه تر کنه ☺

- قطعا نظرات شما نتیجه توجه به آموزش های تولیدی ماست که موجب انگیزه بیشتر برای خلق آموزش ها و مثال ها از سایر مباحث جاوا خواهد شد.

- فروش، ویرایش،کپی این جلسه اموزشی حرام می باشد.

پیروز و موفق باشید

---

# 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.**