جامعة طرابلس
كلية تقنية المعلومات
قسم هندسة البرمجيات

البرمجة المرئية Visual Programming
خريف 2024

المحاضرة الخامسة
Node Styling

مواضيع المحاضرة

- ما هو Cascading Style Sheet؟
- ما هو النمط Style؟
- التعامل مع المصطلحات في javaFX CSS
- اضافة الانماط في javaFX من الخارج
- انشاء ملف CSS
- اضافة الانماط من الداخل Inline Styles
- مقارنة بين النمط الداخلي Inline والخارجي StyleSheet
- أولويات الأنماط عند التعامل مع العقدة nodes
- وراثة خصائص CSS
- انواع خصائص CSS
- المحددات Selectors

2

ما هو Cascading Style Sheet؟

- CSS (ورقة الأنماط المتتالية) هي لغة تستخدم لوصف العرض (الشكل أو النمط) لعناصر واجهة المستخدم في تطبيق واجهة المستخدم الرسومية GUI.
- تم تطوير CSS بشكل أساسي للاستخدام في صفحات الويب لتصميم عناصر HTML.
- تتيح JavaFX تحديد الشكل (أو النمط) لتطبيقات JavaFX باستخدام CSS.
- يمكنك تعريف عناصر واجهة المستخدم باستخدام مكتبات فئة JavaFX أو FXML ويتم استخدام CSS لتحديد مظهرها.

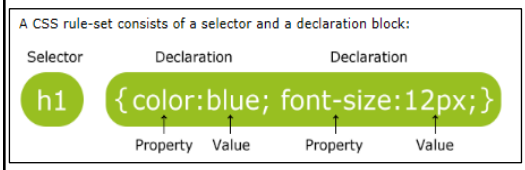
3

ما هو Cascading Style Sheet؟

- توفر CSS صيغة معينة لكتابة القواعد rules التي تطبق على الخصائص المرئية.
- تتكون القاعدة من محدد *selector* ومجموعة من أزواج *property-value*.
  - المحدد *selector* هو سلسلة حرفية تحدد عناصر واجهة المستخدم UI التي سيتم تطبيق القواعد عليها.
  - تتكون *property-value* من: اسم الخاصية، القيمة المقابلة لها مفصولة بنقطتين(:).
  - يتم فصل قيمة الخاصية الأخرى بفاصلة منقوطة (;).
  - يتم وضع مجموعة أزواج الخاصية والقيمة داخل أقواس المجموعة { } مسبوقة بالمحدد.

4

ما هو Cascading Style Sheet؟




```
.button {
        -fx-background-color: red;
        -fx-text-fill: white;
}
```

- الزر Button هو المحدد Selector، حيث يحدد أن القاعدة ستطبق على جميع الأزرار Buttons.
- fx-background-color و fx-text-fill هي أسماء خصائص مع تعيين قيمها بالأحمر والأبيض على التوالي.
- عند تطبيق القاعدة المذكورة أعلاه، سيكون لجميع الأزرار لون خلفية حمراء ولون النص أبيض.

5

ما هو Cascading Style Sheet؟

```
body {
    background-color: lightblue;
}
```

CSS with HTML

```
.root{
    -fx-background-color: lightblue;
}
```

CSS in JavaFX

- تلميح Tip: يشبه استخدام CSS في JavaFX استخدام CSS مع HTML.
- إذا كنت قد استخدمت CSS و HTML من قبل، فستبدو المعلومات الواردة في هذه المحاضرة مألوفة لك. الخبرة السابقة مع CSS ليست ضرورية لفهم كيفية استخدام CSS في JavaFX.

6

ما هو النمط Style؟

- تعرف قاعدة CSS (CSS rule) باسم النمط Style. بينما تُعرف مجموعة قواعد CSS بورقة الأنماط *style sheet*.
- اصطلاحات التسمية في JavaFX CSS تعتمد أسماء فئات نمط CSS على الأسماء البسيطة لفئات JavaFX التي تمثل العقدة node في scene graph.
- جميع أسماء فئات النمط مكتوبة بأحرف صغيرة.

7

التعامل مع المصطلحات في javaFX CSS

- إذا كان اسم الفئة JavaFX يتكون من عدة كلمات، على سبيل المثال، TextField، يتم إدراج واصلة hyphen بين الكلمتين للحصول على اسم فئة النمط.
- على سبيل المثال، تعتبر فئات الأنماط لفئات TextField وCheckBox عبارة عن text-field وcheck-box، على التوالي.

```
.text-field{
    -fx-background-color: blue;
    -fx-font-size: 15px;
```

8

التعامل مع المصطلحات في javaFX

- تبدأ أسماء الخصائص في أنماط JavaFX بـ -fx-،على سبيل المثال يصبح اسم الخاصية font-size في أنماط CSS -fx-font-size
- يتم إدراج واصلة بين كلمتين؛ إذا كانت الخاصية تتكون من عدة كلمات فإنه يحول الاسم إلى أحرف صغيرة ويبدأه بـ -fx-.
- على سبيل المثال ، بالنسبة text Alignment، سيكون اسم الخاصية هو - fx-text-alignment

9

اضافة الانماط في javaFX من الخارج

- يتم اضافة أوراق الأنماط باستخدام getStylesheets() في فئتي Scene و Parent للحصول على المرجع الذي يحتوي على الانماط كما يمكن إضافة عناوين URL.
- ملاحظة Note: تتيح لك وظيفة CSS url() الارتباط بمورد مثل الصورة.

```
// Add a style sheet to the scene
scene.getStylesheets().add("resources/css/buttonstyles.css");
```

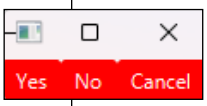
10

styles.css

```
.button {
    -fx-background-color: red;
    -fx-text-fill: white;
}
```

Lect5Ex1.java

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
public class Lect5Ex1 extends Application {
    public static void main(String[] args) {
        Application.launch(args);
    }
    @Override
    public void start(Stage stage) {
        Button yesBtn = new Button("Yes");
        Button noBtn = new Button("No");
        Button cancelBtn = new Button("Cancel");
        HBox root = new HBox();
        root.getChildren().addAll(yesBtn, noBtn, cancelBtn);
        Scene scene = new Scene(root);
        // Add a style sheet to the scene
        scene.getStylesheets().add("styles.css");
        stage.setScene(scene);
        stage.setTitle("Styling Buttons");
        stage.show();
    }
}
```



مثال

11

اضافة الانماط في javaFX من الخارج

- إذا كنت تواجه مشكلات في الوصول إلى أوراق الأنماط الخاصة بك باستخدام التقنية المذكورة سابقا، فيمكنك استخدام عناوين URL. يمكنك أيضًا استخدام getResource() أو ClassLoader للحصول على عنوان URL الخاص بورقة الأنماط الخاصة بك.

```
scene.getStylesheets().add(getClass().getResource("l5css.css")
        .toExternalForm());

String urlString = Test.class.getClassLoader()
                   .getResource("resources/css/hjfx.css")
                   .toExternalForm();
scene.getStylesheets().add(urlString);
```

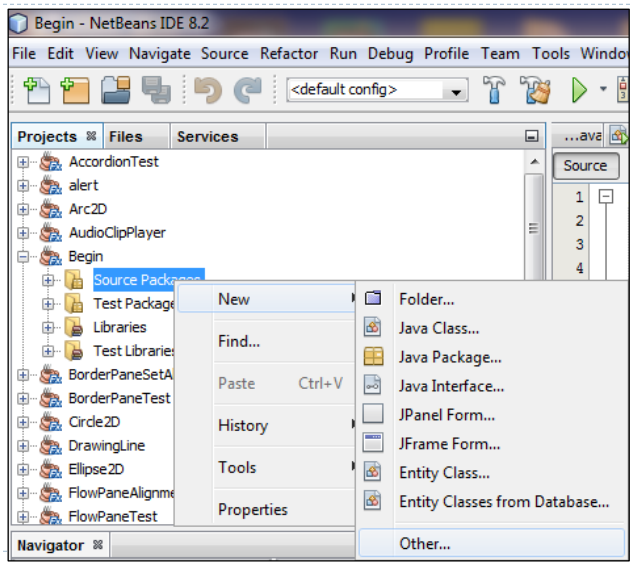
12

انشاء ملف CSS

- على سبيل المثال، لأنشاء ملف CSS في نافذة مشاريع NetBeans IDE، قم بفتح مشروع موجود مسبقا مثلا مشروع باسم Begin، ثم قم بالضغط على علامة الزائد الموجودة بجوار اسم المشروع، فيظهر لك مجلد **Source Packages**، قم بالضغط بالزر الايمن للفارة على المجلد ثم اختار New وخيار Other. أنظر الشكل التالي:

13

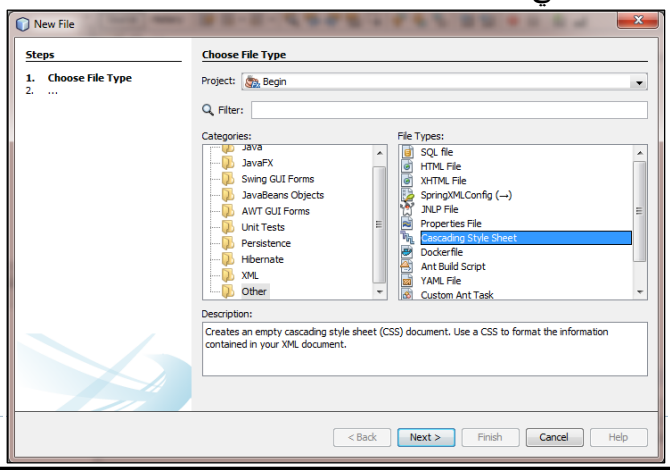
انشاء ملف CSS



14

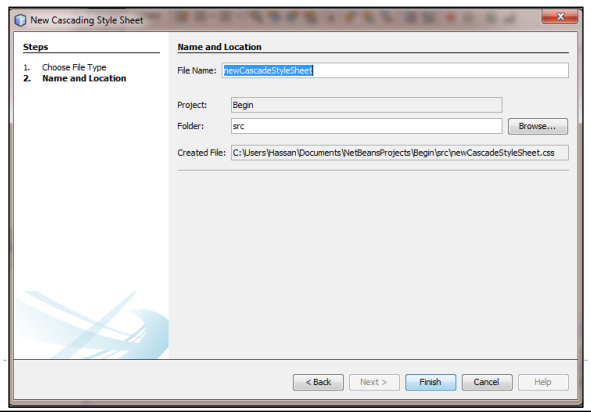
انشاء ملف CSS

- تظهر لك نافذة جديدة تحت خيار Categories اضغط على Other وتحت File Types اختار Cascading style sheet ثم اضغط Next. أنظر الشكل التالي:



انشاء ملف CSS

- تظهر لك نافذة جديدة تحتوي على اسم ملف CSS، والمشروع و مجلد الذي سيحتوي على ملف CSS، تستطيع التأكد من أن المجلد يوجد بداخل مجلد اسم المشروع بالضغط على Browse. يجب أن يكون ملف CSS تحت مجلد src\Begin. بعد التأكد اضغط على finish. أنظر الشكل التالي:



اضافة الانماط من الداخل Inline Styles

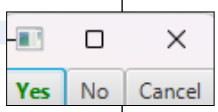
- قد تأتي أنماط CSS الخاصة بالعقدة Node في scene graph من أوراق أنماط الخارجية أو نمط داخلي.
- فئة العقدة Node class لها خاصية من نوع String.
- يتم وضع النمط الداخلي داخل خاصية العقدة.
- يمكن استخدام setStyle() لضبط النمط الداخلي للعقدة.

17

مثال

Lect5Ex1.java

```
1  import javafx.application.Application;
2  import javafx.scene.Scene;
3  import javafx.scene.control.Button;
4  import javafx.scene.layout.HBox;
5  import javafx.stage.Stage;
6  public class Lect5Ex1 extends Application {
7      public static void main(String[] args) {
8          Application.launch(args);
9      }
10     @Override
11     public void start(Stage stage) {
12         Button yesBtn = new Button("Yes");
13         Button noBtn = new Button("No");
14         Button cancelBtn = new Button("Cancel");
15         HBox root = new HBox();
16         root.getChildren().addAll(yesBtn, noBtn, cancelBtn);
17         Scene scene = new Scene(root);
18         yesBtn.setStyle("-fx-text-fill:green; -fx-font-weight: bold;");
19         stage.setScene(scene);
20         stage.setTitle("Styling Buttons");
21         stage.show();
22     }
23 }
```



18

مقارنة بين النمط الداخلي Inline والخارجي StyleSheet

- **ورقة الأنماط الخارجي StyleSheet:**
  - يتكون النمط الخارجي من محدد selector وقيمة الخاصية property-value
  - يؤثر على أكثر من عقد في scene graph.
  - يعتمد عدد العقد المتأثرة بناء على عدد العقد التي تطابق محدد النمط.
- **النمط الداخلي Inline :**
  - لا يحتوي النمط الداخلي على محدد selector.
  - يتألف من قيمة الملكية محددة فقط property-value.
  - يؤثر النمط الداخلي على العقدة التي تم تعيينه عليها.

19

أولويات الأنماط عند التعامل مع العقدة nodes

- في تطبيق JavaFX، من الممكن ، أن تأتي الخصائص المرئية للعقد من مصادر متعددة.
- على سبيل المثال:
  - يمكن ضبط حجم خط الزر أثناء وقت التشغيل JavaFX runtime.
  - يمكن إضافة أوراق الأنماط إلى Parent وScene.
  - يمكن استخدام النمط الداخلي للزر.
  - برمجيا يمكن استخدام setFont(Font f) للتعديل في الخط.

20

مثال

- من الكود البرمجي التالي، ماذا سيكون حجم خط الزر؟
- هل سيكون حجم الخط الافتراضي الذي تم تعيينه أثناء وقت التشغيل JavaFX runtime، 24بكسل ،أو عن طريق النمط الخارجي في الملف Stylespriorities.css، أو 16بكسل الذي تم تعيينه بواسطة النمط الداخلي، أو 10بكسل الذي تم تعيينه بواسطة البرنامج باستخدام setFont()؟

```
Button yesBtn = new Button("Yes");
yesBtn.setStyle("-fx-font-size: 16px");
yesBtn.setFont(new Font(10));

Scene scene = new Scene(yesBtn);
scene.getStylesheets().addAll("resources/css/stylespriorities.css");
...
```

???

The Content of stylespriorities.css File

```
.button {
        -fx-font-size: 24px;
        -fx-font-weight: bold;
}
```

أولويات الأنماط عند التعامل مع العقدة nodes

- تستخدم JavaFX أثناء وقت تشغيل قواعد الأولوية التالية لتعيين الخصائص المرئية للعقدة.
- يتم استخدام المصدر ذي الأولوية الأعلى والذي يحتوي على قيمة للخاصية:
  - النمط الداخلي Inline له (الأولوية العليا)
  - أوراق النمط الخارجي الخاصة بالاب Parent.
  - أوراق النمط الخارجي الخاصة بالمشهد Scene.
  - القيم الموجودة في التعليمات البرمجية باستخدام JavaFX API.
- ملاحظة من الخطأ الشائع تعيين نفس خصائص إلى العقدة في ورقة الأنماط وكذلك في التعليمات البرمجية باستخدام Java API. في هذه الحالة ، تفوز الأنماط الموجودة في ورقة الأنماط ويقضي المطورون ساعات لا تحصى في محاولة للعثور على أسباب عدم تفعيل الخصائص المحددة في الكود.

وراثة خصائص CSS

- تقدم JavaFX نوعين من التوريث لخصائص CSS:
  - وراثة أنواع خصائص CSS.
  - وراثة قيم خصائص CSS.

- تلميح Tip: يمكن تحديد التوريث كقيمة لخاصية CSS للعقدة إذا كنت تريد أن يتم توريث القيمة من الاب parent.

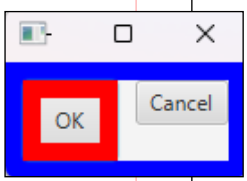
23

مثال

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;

public class Lect5Ex2 extends Application {

    public static void main(String[] args) {
    Application.launch(args);
}
@Override
public void start(Stage stage) {
    Button okBtn = new Button("OK");
    Button cancelBtn = new Button("Cancel");
    HBox root = new HBox(10); // 10px spacing
    root.getChildren().addAll(okBtn, cancelBtn);
    // Set styles for the OK button and its parent HBox
    root.setStyle("-fx-cursor: hand;-fx-border-color: blue;-fx-border-width: 10px;");
    okBtn.setStyle("-fx-border-color: red;-fx-border-width: inherit;");
    Scene scene = new Scene(root);
    stage.setScene(scene);
    stage.setTitle("CSS Inheritance");
    stage.show();
}
}
```



24

انواع خصائص CSS

- جميع القيم في Java وفي JavaFX لها نوع.
- قيم خصائص CSS التي تم تعيينها في الأنماط لها أنواع أيضًا.
- كل نوع من القيم له بناء جملة مختلفة.
- تدعم JavaFX CSS الأنواع التالية:
  - **inherit**
  - **boolean**
  - **string**
  - **number**
  - **URL**
  - **font**

25

انواع خصائص CSS

- **نوع الوراثة inherit**
  - يتم استخدامه لوراثة قيمة خاصية CSS للعقدة من أباها.

```
// Set styles for the OK button and its parent HBox
root.setStyle("-fx-cursor: hand;-fx-border-color: blue;-fx-border-width: 5px;");
okBtn.setStyle("-fx-border-color: red;-fx-border-width: inherit;");
```

- **النوع المنطقي boolean**
  - يمكنك تحديد قيم النوع المنطقي على أنها true or false.
  - يمكن أيضًا تحديدها كسلاسل حرفية: "true" or "false"

```
.text-field {
        -fx-display-caret: false;
}
```

26

انواع خصائص CSS

- **نوع السلسلة String**
  - يمكن إحاطة قيم السلسلة بعلامات اقتباس فردية أو علامات اقتباس مزدوجة.

```
-fx-font: normal bold 20px 'serif';
```

- **النوع URL**
- يمكن تحديد URL باستخدام وظيفة url(<address>) حيث يتم استخدام <العنوان> لتحديد موقع ملف CSS أو صورة معينة:

```
.root{
    -fx-background-image: url(Photo.jpeg);
}
```

27

انواع خصائص CSS

- **النوع الرقم Number**
  - يمكن تمثيل القيم الرقمية كأعداد صحيحة أو أعداد حقيقية. يتم تحديدها باستخدام تنسيق الأرقام العشري. يضبط النمط التالي 0.60 نسبة عالية من التظليم.

```
.my-style {
        -fx-opacity: 0.60;
}
```

- **نوع الخط**
  - يتكون الخط من أربع خصائص: family, size, style, weight.
  - هناك طريقتان لتحديد الخط Font للخاصية:
  - حدد الخصائص الأربع للخط بشكل منفصل باستخدام خصائص CSSالأربعة: fx-font-family و fx-font-size و fx-font-style وfx-font-weight.

```
.button{

    -fx-font-family: "serif";
    -fx-font-size: 20px;
    -fx-font-style: normal;
    -fx-font-weight: bolder;
}
```

28

انواع خصائص CSS

- استخدم خاصية CSS المختصرة fx-font لتحديد كل الخصائص الأربع كقيمة واحدة لكن بترتيب محدد:

font-style | font-variant | font-weight | font-size | font-family

```
-fx-font: italic bold 20px arial;
```

29

ما هو المحدد Selector

- يدعم JavaFX CSS عدة أنواع من المحددات مثل محددات الفئة ومحددات المعرفات والمحددات العامة.
- استخدام محددات الفئة يقوم محدد فئة النمط بتطبيق النمط المرتبط على كل العقد ، والتي لها نفس اسم فئة النمط مثل اسم المحدد.
  - استخدام المحدد Class Selector
  - استخدام المحدد ID Selector
  - الدمج بين المحددين ID Selector & Class
  - استخدام المحدد العام *

30

استخدام المحدد Class Selector

- تطبيق النمط المرتبط على كل العقد، والتي لها نفس اسم فئة النمط مثل اسم المحدد.

```
.hbox {
        -fx-border-color: blue;
        -fx-border-width: 2px;
        -fx-border-radius: 5px;
        -fx-border-insets: 5px;
        -fx-padding: 10px;
        -fx-spacing: 5px;
        -fx-background-color: lightgray;
        -fx-background-insets: 5px;
}

.button {
        -fx-text-fill: blue;
}
```

```
.root {
        -fx-cursor: hand;
        -my-button-color: blue;
}

.button {
        -fx-text-fill: -my-button-color;
}
```

31

استخدام المحدد ID Selector

- تعيين معرف فريد لكل عقدة في الرسم البياني للمشهد. محدد المعرف في ورقة الأنماط مسبوق بعلامة الجنيه (#).

```
Button b1 = new Button("Close");
b1.setId("closeBtn");
```

```
.button {
        -fx-text-fill: blue;
}

#closeButton {
        -fx-text-fill: red;
}
```

32

الدمج بين المحددين ID Selector & Class

- يمكن أن يستخدم المحدد مجموعة من فئة النمط والمعرف. في هذه الحالة، يطابق المحدد جميع العقد مع فئة النمط والمعرف المحدد.

```
#closeButton.button {
        -fx-text-fill: red;
}
```

```
.button#closeButton {
        -fx-text-fill: red;
}
```

33

استخدام المحدد العام *

- أُستخدم علامة النجمة (*) كمحدد عام ، والذي يطابق أي عقدة.

```
* {
        -fx-text-fill: blue;
}
```

34

استخدام عدة محددات معا

- تجميع محددات متعددة إذا كانت نفس خصائص CSS تنطبق على محددات متعددة.
  - يمكنك استخدام أنماط متعددة من خلال تكرار تعريفات الخصائص.

```
.button {
        -fx-text-fill: blue;
}

.label {
        -fx-text-fill: blue;
}
```

  - يمكنك تجميع كل المحددات في نمط واحد ، مع فصل المحددات بفاصلة.

```
.button, .label {
        -fx-text-fill: blue;
}
```

35

نهاية المحاضرة

36