جامعة طرابلس
كلية تقنية المعلومات

البرمجة المرئية
Visual Programming

ITSE423

إعداد: حسن علي حسن إبراهيم
المحاضر: فاطمة علي بن الأشهر

المحاضرة الخامسة

مواضيع المحاضرة

- عقد التصميم: CSSفي JavaFX
- ما هي أوراق الأنماط المتتالية؟
- اصطلاحات التسمية لأنماط أوراق الأنماط المتتالية في JavaFX
- إضافة أوراق الأنماط إلى المشهد
- إضافة أنماط مضمنة للعقدة
- الأنواع المختلفة لخصائص ورقة الأنماط المتتالية
- حول محددات أنماط أوراق الأنماط المتتالية

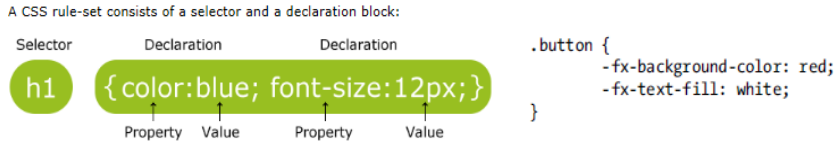
ما هو Cascading Style Sheet CSS؟

- ما هي ورقة الأنماط المتتالية CSS
- CSS 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.

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 معكوفة { } مسبوقة بالمحدد.

ما هو Cascading Style Sheet؟



- 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 على الأحمر والأبيض ، على التوالي.
- عند تطبيق القاعدة المذكورة أعلاه ، سيكون لجميع الأزرار لون خلفية حمراء ولون النص أبيض.

ما هو Cascading Style Sheet؟

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

CSS in JavaFX

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

CSS with HTML

- 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.

ما هو النمط Style؟

- تعرف قاعدة CSS أيضًا باسم النمط Style. 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.

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 TextFieldو CheckBox عبارة عن text-field وcheck-box، على التوالي.

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

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

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");
```

اضافة الانماط في javaFX من الخارج

```
// Add two style sheets, ss1.css and ss2.css to a scene
Scene scene = ...
scene.getStylesheets().addAll("resources/css/ss1.css", "resources/css/ss2.css");

// Add a style sheet, vbox.css, to a VBox (a Parent)
VBox root = new VBox();
root.getStylesheets().add("vbox.css");
```

- مثال:

```
scene.getStylesheets().addAll("lecture5_css/l5css.css","lecture5_css/l5css2.css");
```


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.

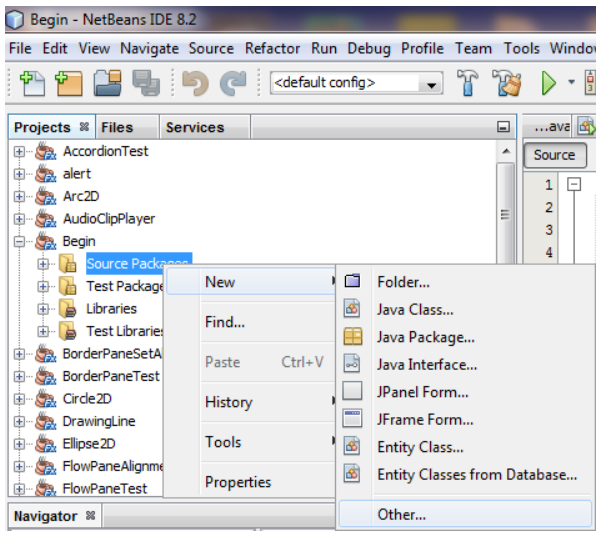
```
scene.getStylesheets().add(getClass().getResource("l5css.css")
        .toExternalForm());

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

انشاء ملف CSS

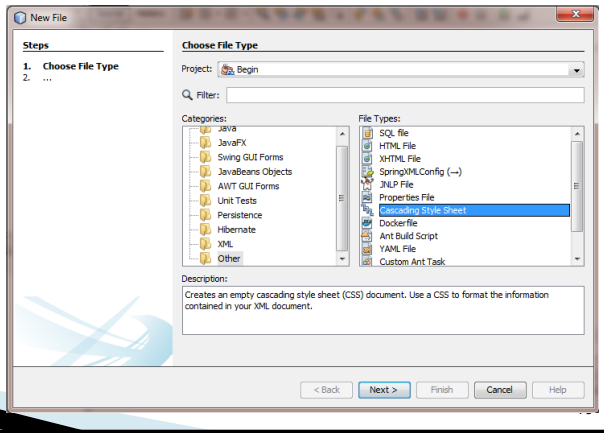
- على سبيل المثال، لأنشاء ملف CSS في نافذة مشاريع NetBeans IDE، قم بفتح مشروع موجود مسبقا مثلا مشروع باسم Begin، ثم قم بالضغط على علامة الزائد الموجودة بجوار اسم المشروع، فيظهر لك مجلد **Source Packages**، قم بالضغط بالزر الايمن للفارة على المجلد ثم اختار New وخيار Other. أنظر الشكل التالي:

انشاء ملف CSS



انشاء ملف CSS

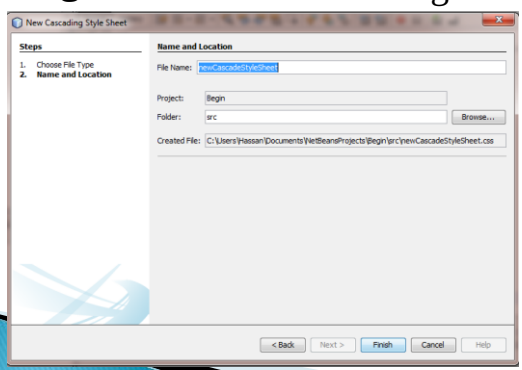
- تظهر لك نافذة جديدة تحت خيار Categories اضغط على Other وتحت File Types اختار Cascading style sheet ثم اضغط Next. أنظر الشكل التالي:



Hassan Ebrahem

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:



Hassan Ebrahem

16

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.

اضافة الانماط من الداخل Inline Styles

- في الجزء البرمجي التالي يستخدم النمط الداخلي لزر button لعرض النص باللون الأحمر ويوكن الخط عريض.
- مثال:

```
Button yesBtn = new Button("Yes");
yesBtn.setStyle("-fx-text-fill:red;      -fx-font-weight: bold;");
```

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.

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.

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;
}
```

- الإجابة الصحيحة هي 16بكسل، والتي يتم تعيينها بواسطة النمط الداخلي.

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 Parentل.
  - 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

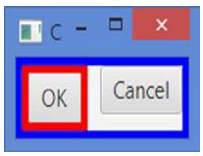
- تقدم JavaFX نوعين من التوريث لخصائص CSS:
- وراثة أنواع خصائص CSS.
- وراثة قيم خصائص CSS.

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

وراثة خصائص CSS

```
/* Parent Node (HBox)*/
-fx-cursor: hand;
-fx-border-color: blue;
-fx-border-width: 5px;

/* Child Node (OK Button)*/
-fx-border-color: red;
-fx-border-width: inherit;
```




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

public class CSSInheritance 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: 5px;");
		okBtn.setStyle("-fx-border-color: red;-fx-border-width: inherit;");

		Scene scene = new Scene(root);
		stage.setScene(scene);
		stage.setTitle("CSS Inheritance");
		stage.show();
	}
}
```

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**
- **URI**
- **font**

```
#btn{
    -fx-background-color: #ff9900;
}

.text1{
    -fx-font-size: 24px ;
}

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


النوع inherit و النوع boolean

- **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;
}
```

نوع السلسلة string

- **نوع السلسلة String**
- يمكن إحاطة قيم السلسلة بعلامات اقتباس فردية أوعلامات اقتباس مزدوجة. إذا كانت قيمة السلسلة محاطة بعلامات اقتباس مزدوجة، وكانت علامة اقتباس مزدوجة كجزء من القيمة، فيتم وضعها كالتالي \" أو \22. وبالمثل ، إذا كانت علامة الاقتباس الفردية كجزء من قيمة السلسلة فيتم وضعها كالتالي ، مثل \' أو \27.

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


النوع URI و النوع number

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

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

- **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 التضليل.

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

النوع الخط font

- **نوع الخط**
- يتكون الخط من أربع خصائص: family, size, style, and weight.
- هناك طريقتان لتحديد الخط Font للخاصية:
- حدد الخصائص الأربع للخط بشكل منفصل باستخدام خصائص CSS الأربعة:-
- fx-font-family و fx-font-size و fx-font-style و fx-font-weight.
- استخدم خاصية CSS المختصرة fx-font لتحديد كل الخصائص الأربع كقيمة واحدة.

النوع الخط font

- fx-font-family و fx-font-size و fx-font-style و fx-font-weight.

```
.button{

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

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

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

ماهو المحدد Selector

- فهم محددات النمطيحتوي كل نمط في ورقة الأنماط على محدد مرتبط يحدد العقد في الرسم البياني للمشهد التي يتم تطبيق قيم خصائص JavaFX CSSالمرتبطة عليها. يدعم JavaFX CSS عدة أنواع من المحددات مثل محددات الفئة ومحددات المعرفات والمحددات العامة.
- استخدام محددات الفئةيقوم محدد فئة النمط بتطبيق النمط المرتبط على كل العقد ، والتي لها نفس اسم فئة النمط مثل اسم المحدد.

استخدام المحدد Class Selector

استخدام محددات الفئةيقوم محدد فئة النمط بتطبيق 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;
}
```

استخدام المحدد ID Selector

- استخدام محددات الهويةتحتوي فئة العقدة على خاصية معرف من نوع StringProperty، والتي يمكن استخدامها لتعيين معرف فريد لكل عقدة في الرسم البياني للمشهد.محدد المعرف في ورقة الأنماط مسبوق بعلامة الجنيه (#).

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

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

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


الدمج بين المحددين ID Selector & Class

- الجمع بين معرف ومحدد الفئةيمكن أن يستخدم المحدد مجموعة من فئة النمط والمعرف. في هذه الحالة ، يطابق المحدد جميع العقد مع فئة النمط والمعرف المحدد. ضع في اعتبارك النمط التالي:

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

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

المحدد العام *

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

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


استخدام عدة محددات معا

- تجميع محددات متعددةإذا كانت نفس خصائص CSSتنطبق على محددات متعددة ، فلديك خياران:يمكنك استخدام أنماط متعددة من خلال تكرار تعريفات الخصائص.يمكنك تجميع كل المحددات في نمط واحد ، مع فصل المحددات بفاصلة.

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

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

ملخص المحاضرة

- في المثال التالي قمنا بإنشاء برنامج إسمه Main, موجود بداخل مجدل إسمه main, مهمته فقط طباعة الجملة "Hello World!" عند التشغيل.

نهاية المحاضرة