Release: JavaFX 2.2
Last Updated: September 2013
Download as PDF

[+] Show/Hide Table of Contents

[+] Show/Hide Application Files

Profiles

Alla Redko
Technical Writer, Oracle

Alla is a technical writer for Oracle. She lives in St. Petersburg, Russia, and develops tutorials and technical articles for Java and JavaFX technologies. Prior to her assignment at Oracle, she worked as a technical writer in different IT companies.

22 Menu

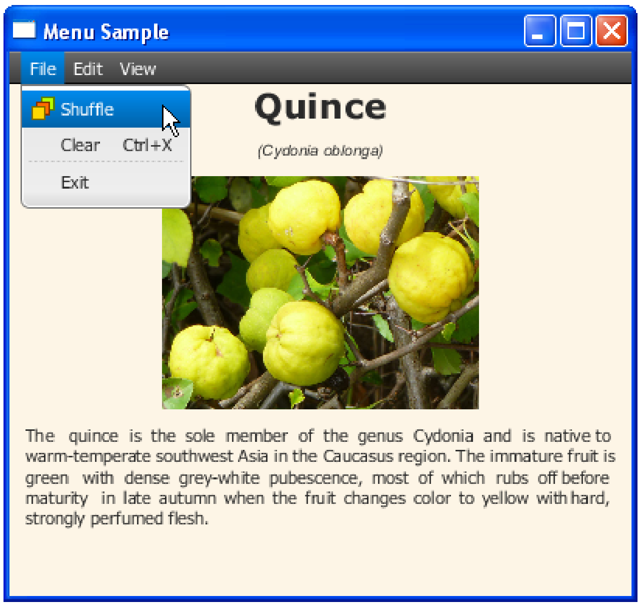
This chapter explains how to create menus and menu bars, add menu items, group the menus into categories, create submenus, and set context menus.

You can use the following classes of the JavaFX API to build menus in your JavaFX application.

- MenuBar
- MenuItem
  - Menu
  - CheckMenuItem
  - RadioMenuItem
  - Menu
  - CustomMenuItem
    - SeparatorMenuItem
- ContextMenu

Figure 22-1 shows a screen capture of an application with a typical menu bar.

Figure 22-1 Application with a Menu Bar and Three Menu Categories



The quince is the sole member of the genus Cydonia and is native to warm-temperate southwest Asia in the Caucasus region. The immature fruit is green with dense grey-white pubescence, most of which rubs off before maturity in late autumn when the fruit changes color to yellow with hard, strongly perfumed flesh.

Description of "Figure 22-1 Application with a Menu Bar and Three Menu Categories"

Building Menus in JavaFX Applications

A menu is a list of actionable items that can be displayed upon a user's request. When a menu is visible, users can select one menu item at time. After a user clicks an item, the menu returns to the hidden mode. By using menus, you can save space in your application user interface (UI) by placing in menus the functionality that does not always need to be visible.

The menus in a menu bar are typically grouped into categories. The coding pattern is to declare a menu bar, define the category menus, and populate the category menus with menu items. Use the following menu item classes when building menus in your JavaFX applications:

- `MenuItem` – to create one actionable option
- `Menu` – to create a submenu
- `RadioButtonItem` – to create a mutually exclusive selection
- `CheckMenuItem` – to create an option that can be toggled between selected and unselected states

To separate menu items within one category, use the `SeparatorMenuItem` class.

The menus organized by categories in a menu bar are typically located at the top of the window, leaving the rest of the scene for crucial UI elements. If, for some reasons, you cannot allot any visual part of your UI for a menu bar, you can use context menus that the user opens with a mouse click.

Creating a Menu Bar

Although a menu bar can be placed elsewhere in the user interface, typically it is located at the top of the UI and it contains one or more menus. The menu bar automatically resizes to fit the width of the application window. By default, each menu added to the menu bar is represented by a button with the text value.

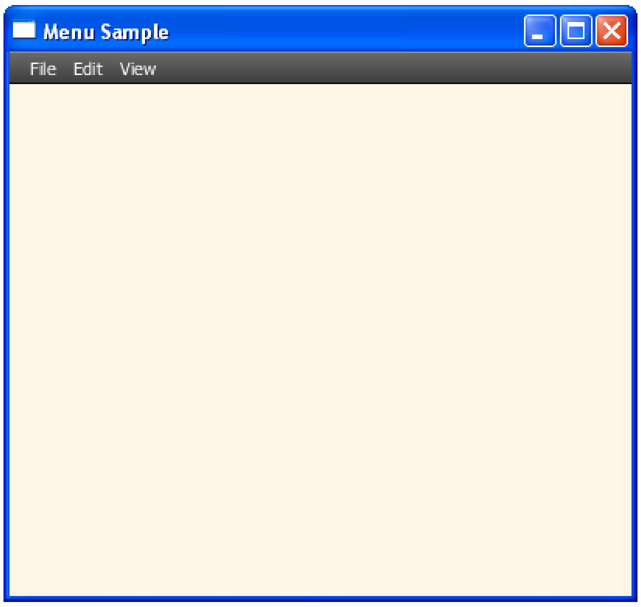
Consider an application that renders reference information about plants such as their name, binomial name, picture, and a brief description. You can create three menu categories: File, Edit, and View, and populate them with the menu items. Example 22-1 shows the source code of such an application with the menu bar added.

Example 22-1 Menu Sample Application

```
import java.util.AbstractMap.SimpleEntry;
import java.util.Map.Entry;
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.effect.DropShadow;
import javafx.scene.effect.Effect;
import javafx.scene.effect.Glow;
import javafx.scene.effect.SepiaTone;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.stage.Stage;

public class MenuSample extends Application {

    final PageData[] pages = new PageData[] {
        new PageData("Apple",
            "The apple is the pomaceous fruit of the apple tree, species Malus "
            + "domestica in the rose family (Rosaceae). It is one of the most "
            + "widely cultivated tree fruits, and the most widely known of "
            + "the many members of genus Malus that are used by humans. "
            + "The tree originated in Western Asia, where its wild ancestor, "
            + "the Alma, is still found today.",
            "Malus domestica"),
        new PageData("Hawthorn",
            "The hawthorn is a large genus of shrubs and trees in the rose "
            + "family, Rosaceae, native to temperate regions of the Northern "
            + "Hemisphere in Europe, Asia and North America. "
            + The name hawthorn was "
            + "originally applied to the species native to northern Europe, "
            + "especially the Common Hawthorn C. monogyna, and the unmodified "
            + "name is often so used in Britain and Ireland.",
            "Crataegus monogyna"),
        new PageData("Ivy",
            "The ivy is a flowering plant in the grape family (Vitaceae) native to
            + " eastern Asia in Japan, Korea, and northern and eastern China. "
            + "It is a deciduous woody vine growing to 30 m tall or more given "
            + "suitable support,  attaching itself by means of numerous small "
            + "branched tendrils tipped with sticky disks.",
            "Parthenocissus tricuspidata"),
        new PageData("Quince",
            "The quince is the sole member of the genus Cydonia and is native to "
            + "warm-temperate southwest Asia in the Caucasus region. The "
            + "immature fruit is green with dense grey-white pubescence, most "
            + "of which rubs off before maturity in late autumn when the fruit "
            + "changes color to yellow with hard, strongly perfumed flesh.",
            "Cydonia oblonga")
    };

    final String[] viewOptions = new String[] {
        "Title",
        "Binomial name",
        "Picture",
        "Description"
    };

    final Entry<String, Effect>[] effects = new Entry[] {
        new SimpleEntry<String, Effect>("Sepia Tone", new SepiaTone()),
        new SimpleEntry<String, Effect>("Glow", new Glow()),
        new SimpleEntry<String, Effect>("Shadow", new DropShadow())
    };

    final ImageView pic = new ImageView();
    final Label name = new Label();
    final Label binName = new Label();
    final Label description = new Label();

    public static void main(String[] args) {
        launch(args);
    }

    @Override
    public void start(Stage stage) {
        stage.setTitle("Menu Sample");
        Scene scene = new Scene(new VBox(), 400, 350);
        scene.setFill(Color.OLDLACE);

        MenuBar menuBar = new MenuBar();

        // --- Menu File
        Menu menuFile = new Menu("File");

        // --- Menu Edit
        Menu menuEdit = new Menu("Edit");

        // --- Menu View
        Menu menuView = new Menu("View");

        menuBar.getMenus().addAll(menuFile, menuEdit, menuView);


        ((VBox) scene.getRoot()).getChildren().addAll(menuBar);

        stage.setScene(scene);
        stage.show();
    }


    private class PageData {
        public String name;
        public String description;
        public String binNames;
        public Image image;
        public PageData(String name, String description, String binNames) {
            this.name = name;
            this.description = description;
            this.binNames = binNames;
            image = new Image(getClass().getResourceAsStream(name + ".jpg"));
        }
    }
}
```

Unlike other UI Controls, the `Menu` class and other extensions of the `MenuItem` class do not extend the `Node` class. They cannot be added directly to the application scene and remain invisible until added to the menu bar through the `getMenus` method.

Figure 22-2 Menu Bar is Added to the Application



Description of "Figure 22-2 Menu Bar is Added to the Application"

You can navigate through the menus by using the arrow keys of the keyboard. However, when you select a menu, no action is performed, because the behavior for the menus is not defined yet.

Adding Menu Items

Set the functionality for the File menu by adding the following items:

- Shuffle – to load reference information about plants
- Clear – to remove the reference information and clear the scene
- Separator – to detach menu items
- Exit – to exit the application

Bold lines in Example 22-2 create a Shuffle menu by using the `MenuItem` class and add graphical components to the application scene. The `MenuItem` class enables creating an actionable item with text and graphics. The action performed on a user click is defined by the `setOnAction` method, similar to the `Button` class.

Example 22-2 Adding the Shuffle Menu Item with Graphics

```
import java.util.AbstractMap.SimpleEntry;
import java.util.Map.Entry;
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.effect.DropShadow;
import javafx.scene.effect.Effect;
import javafx.scene.effect.Glow;
import javafx.scene.effect.SepiaTone;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.scene.text.TextAlignment;
import javafx.stage.Stage;

public class MenuSample extends Application {

    final PageData[] pages = new PageData[] {
        new PageData("Apple",
            "The apple is the pomaceous fruit of the apple tree, species Malus "
            +"domestica in the rose family (Rosaceae). It is one of the most "
            +"widely cultivated tree fruits, and the most widely known of "
            +"the many members of genus Malus that are used by humans. "
            +"The tree originated in Western Asia, where its wild ancestor, "
            +"the Alma, is still found today.",
            "Malus domestica"),
        new PageData("Hawthorn",
            "The hawthorn is a large genus of shrubs and trees in the rose "
            + "family, Rosaceae, native to temperate regions of the Northern "
            + "Hemisphere in Europe, Asia and North America. "
            + "The name hawthorn was "
            + "originally applied to the species native to northern Europe, "
            + "especially the Common Hawthorn C. monogyna, and the unmodified "
            + "name is often so used in Britain and Ireland.",
            "Crataegus monogyna"),
        new PageData("Ivy",
            "The ivy is a flowering plant in the grape family (Vitaceae) native"
            +" to eastern Asia in Japan, Korea, and northern and eastern China."
            +" It is a deciduous woody vine growing to 30 m tall or more given "
            +"suitable support,  attaching itself by means of numerous small "
            +"branched tendrils tipped with sticky disks.",
            "Parthenocissus tricuspidata"),
        new PageData("Quince",
            "The quince is the sole member of the genus Cydonia and is native"
            +" to warm-temperate southwest Asia in the Caucasus region. The "
            +"immature fruit is green with dense grey-white pubescence, most "
            +"of which rubs off before maturity in late autumn when the fruit "
            +"changes color to yellow with hard, strongly perfumed flesh.",
            "Cydonia oblonga")
    };

    final String[] viewOptions = new String[] {
        "Title",
        "Binomial name",
        "Picture",
        "Description"
    };

    final Entry<String, Effect>[] effects = new Entry[] {
        new SimpleEntry<String, Effect>("Sepia Tone", new SepiaTone()),
        new SimpleEntry<String, Effect>("Glow", new Glow()),
        new SimpleEntry<String, Effect>("Shadow", new DropShadow())
    };

    final ImageView pic = new ImageView();
    final Label name = new Label();
    final Label binName = new Label();
    final Label description = new Label();
    private int currentIndex = -1;

    public static void main(String[] args) {
        launch(args);
    }
    @Override
    public void start(Stage stage) {
        stage.setTitle("Menu Sample");
        Scene scene = new Scene(new VBox(), 400, 350);
        scene.setFill(Color.OLDLACE);

        name.setFont(new Font("Verdana Bold", 22));
        binName.setFont(new Font("Arial Italic", 10));
        pic.setFitHeight(150);
        pic.setPreserveRatio(true);
        description.setWrapText(true);
        description.setTextAlignment(TextAlignment.JUSTIFY);

        shuffle();

        MenuBar menuBar = new MenuBar();

        final VBox vbox = new VBox();
        vbox.setAlignment(Pos.CENTER);
        vbox.setSpacing(10);
        vbox.setPadding(new Insets(0, 10, 0, 10));
        vbox.getChildren().addAll(name, binName, pic, description);

        // --- Menu File
        Menu menuFile = new Menu("File");
        MenuItem add = new MenuItem("Shuffle",
            new ImageView(new Image("menusample/new.png")));
        add.setOnAction(new EventHandler<ActionEvent>() {
            public void handle(ActionEvent t) {
                shuffle();
                vbox.setVisible(true);
            }
        });

        menuFile.getItems().addAll(add);

        // --- Menu Edit
        Menu menuEdit = new Menu("Edit");

        // --- Menu View
        Menu menuView = new Menu("View");

        menuBar.getMenus().addAll(menuFile, menuEdit, menuView);
        ((VBox) scene.getRoot()).getChildren().addAll(menuBar, vbox);
        stage.setScene(scene);
        stage.show();
    }

    private void shuffle() {
        int i = currentIndex;
        while (i == currentIndex) {
            i = (int) (Math.random() * pages.length);
        }
        pic.setImage(pages[i].image);
        name.setText(pages[i].name);
        binName.setText("(" + pages[i].binNames + ")");
        description.setText(pages[i].description);
        currentIndex = i;
    }


    private class PageData {
        public String name;
        public String description;
        public String binNames;
        public Image image;
        public PageData(String name, String description, String binNames) {
            this.name = name;
            this.description = description;
            this.binNames = binNames;
            image = new Image(getClass().getResourceAsStream(name + ".jpg"));
        }
    }
}
```

When a user selects the Shuffle menu item, the `shuffle` method called within `setOnAction` specifies the title, the binomial name, a picture of the plant, and its description by calculating the index of the elements in the corresponding arrays.

The Clear menu item is used to erase the application scene. You can implement this by making the `VBox` container with the GUI elements invisible as shown in Example 22-3.

Example 22-3 Creating the Clear Menu Item with Accelerator

```
MenuItem clear = new MenuItem("Clear");
    clear.setAccelerator(KeyCombination.keyCombination("Ctrl+X"));
    clear.setOnAction(new EventHandler<ActionEvent>() {
        public void handle(ActionEvent t) {
            vbox.setVisible(false);
        }
});
```

Implementation of the `MenuItem` class enables developers to set a menu accelerator, a key combination that performs the same action as the menu item. With the Clear menu, users can either select the action from the File menu category or press Control Key and X key simultaneously.

The Exit menu closes the application window. Set `System.exit(0)` as an action for this menu item as shown in Example 22-4.

Example 22-4 Creating the Exit Menu Item

```
MenuItem exit = new MenuItem("Exit");
exit.setOnAction(new EventHandler<ActionEvent>() {
    public void handle(ActionEvent t) {
        System.exit(0);
    }
});
```

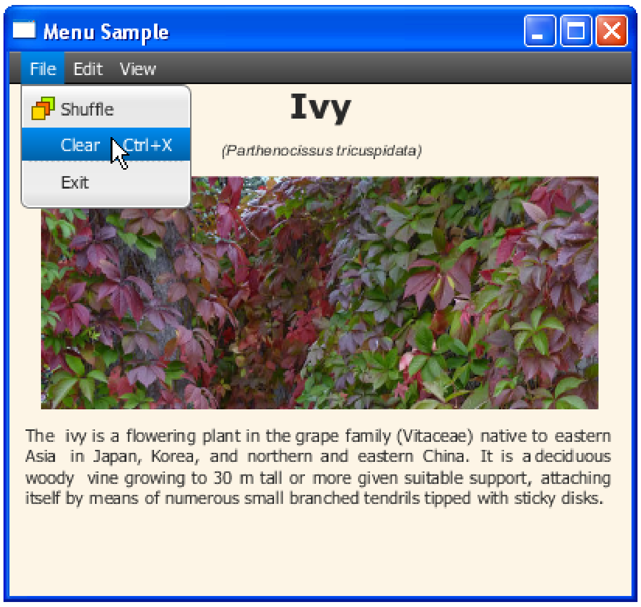
Use the `getItems` method shown in Example 22-5 to add the newly created menu items to the File menu. You can create a separator menu item and add it within the `getItems` method to visually detach the Exit menu item.

Example 22-5 Adding Menu Items

```
menuFile.getItems().addAll(add, clear, new SeparatorMenuItem(), exit);
```

Add Example 22-2, Example 22-3, Example 22-4, and Example 22-5 to the Menu Sample application, and then compile and run it. Select the Shuffle menu item to load reference information about different plants. Then clear the scene (Clear), and close the application (Exit). Figure 22-3 shows selection of the Clear menu item.

Figure 22-3 File Menu with Three Menu Items



Ivy

(Parthenocissus tricuspidata)

The ivy is a flowering plant in the grape family (Vitaceae) native to eastern Asia in Japan, Korea, and northern and eastern China. It is a deciduous woody vine growing to 30 m tall or more given suitable support, attaching itself by means of numerous small branched tendrils tipped with sticky disks.

Description of "Figure 22-3 File Menu with Three Menu Items"

With the View menu, you can hide and show elements of reference information. Implement the `createMenuItem` method and call it within the `start` method to create four `CheckMenuItem` objects. Then add newly created check menu items to the View menu to toggle visibility of the title, binomial name, picture of the plant, and its description. Example 22-6 shows two code fragments that implement these tasks.

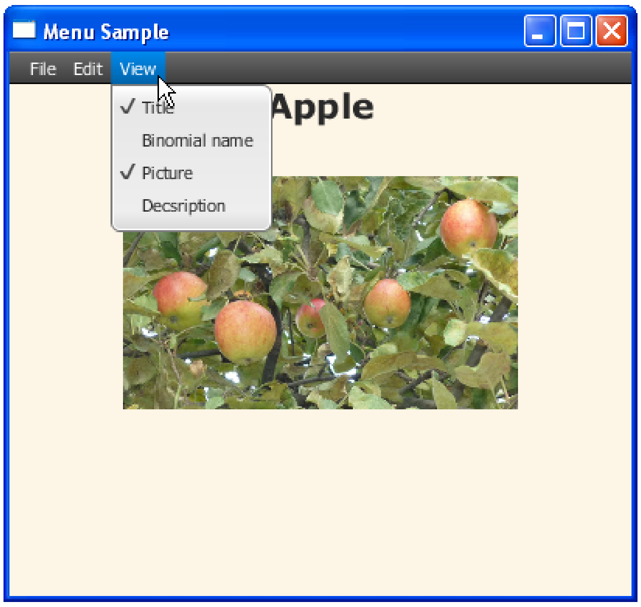
Example 22-6 Applying the CheckMenuItem Class to Create Toggle Options

```
// --- Creating four check menu items within the start method
CheckMenuItem titleView = createMenuItem ("Title", name);
CheckMenuItem binNameView = createMenuItem ("Binomial name", binName);
CheckMenuItem picView = createMenuItem ("Picture", pic);
CheckMenuItem descriptionView = createMenuItem ("Description", description);
menuView.getItems().addAll(titleView, binNameView, picView, descriptionView);

...

// The createMenuItem method
private static CheckMenuItem createMenuItem (String title, final Node node){
    CheckMenuItem cmi = new CheckMenuItem(title);
    cmi.setSelected(true);
    cmi.selectedProperty().addListener(new ChangeListener<Boolean>() {
        public void changed(ObservableValue ov,
        Boolean old_val, Boolean new_val) {
            node.setVisible(new_val);
        }
    });
    return cmi;
}
```

The `CheckMenuItem` class is an extension of the `MenuItem` class. It can be toggled between selected and deselected states. When selected, a check menu item shows a check mark.

Example 22-6 creates four `CheckMenuItem` objects and processes the changing of their `selectedProperty` property. When, for example, a user deselects the `picView` item, the `setVisible` method receives the `false` value, the picture of the plant becomes invisible. When you add this code fragment to the application, compile, and run the application, you can experiment with selecting and deselecting the menu items. Figure 22-4 shows the application in the moment when the title and picture of the plant are shown, but its binomial name and description are hidden.

Figure 22-4 Using Check Menu Items



Description of "Figure 22-4 Using Check Menu Items"

Creating Submenus

For the Edit menu, define two menu items: Picture Effect and No Effects. The Picture Effect menu item is designed as a submenu with three items to set one of the three available visual effects. The No Effects menu item removes the selected effect and restores the initial state of the image.

Use the `RadioMenuItem` class to create the items of the submenu. Add the radio menu buttons to a toggle group to make the selection mutually exclusive. Example 22-7 implements these tasks.

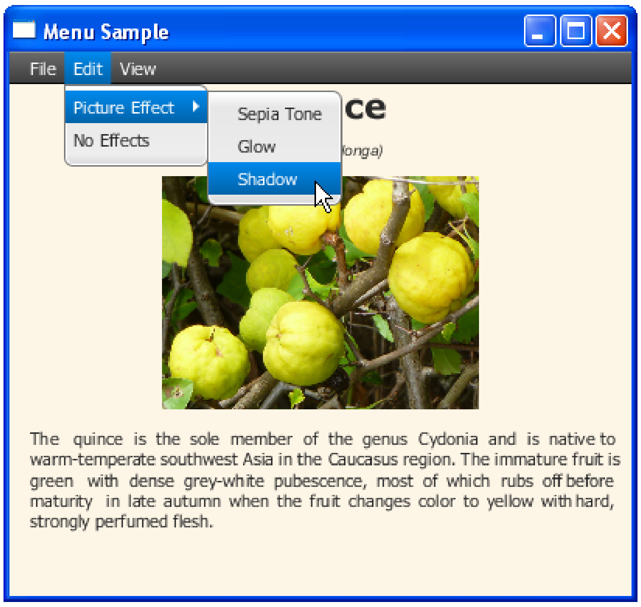
Example 22-7 Creating a Submenu with Radio Menu Items

```
//Picture Effect menu
Menu menuEffect = new Menu("Picture Effect");
final ToggleGroup groupEffect = new ToggleGroup();
for (Entry<String, Effect> effect : effects) {
    RadioMenuItem itemEffect = new RadioMenuItem(effect.getKey());
    itemEffect.setUserData(effect.getValue());
    itemEffect.setToggleGroup(groupEffect);
    menuEffect.getItems().add(itemEffect);
}
//No Effects menu
final MenuItem noEffects = new MenuItem("No Effects");

 noEffects.setOnAction(new EventHandler<ActionEvent>() {
     public void handle(ActionEvent t) {
         pic.setEffect(null);
         groupEffect.getSelectedToggle().setSelected(false);
     }
});

//Processing menu item selection
groupEffect.selectedToggleProperty().addListener(new ChangeListener<Toggle>() {
    public void changed(ObservableValue<? extends Toggle> ov,
        Toggle old_toggle, Toggle new_toggle) {
            if (groupEffect.getSelectedToggle() != null) {
                Effect effect =
                    (Effect) groupEffect.getSelectedToggle().getUserData();
                pic.setEffect(effect);
            }
        }
 });
//Adding items to the Edit menu
menuEdit.getItems().addAll(menuEffect, noEffects);
```

The `setUserData` method defines a visual effect for a particular radio menu item. When one of the items in the toggle group is selected, the corresponding effect is applied to the picture. When the No Effects menu item is selected, the `setEffect` method specifies the `null` value and no effects are applied to the picture.

Figure 22-5 captures a moment when a user is selecting a Shadow menu item.

Figure 22-5 Submenu with Three Radio Menu Items

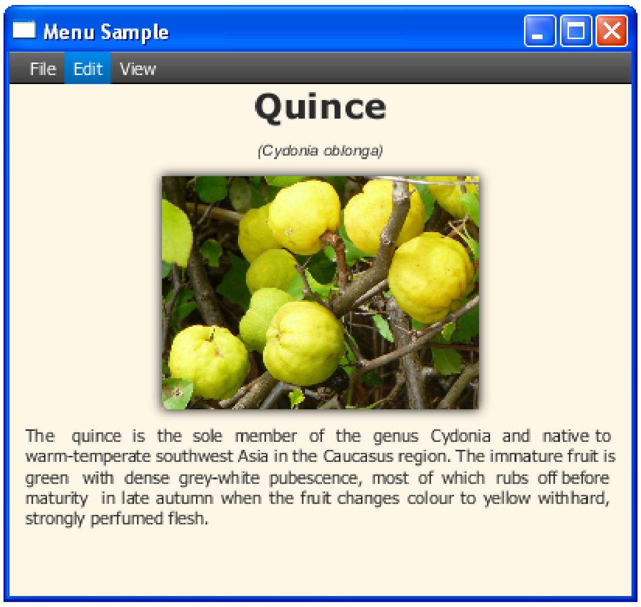


The quince is the sole member of the genus Cydonia and is native to warm-temperate southwest Asia in the Caucasus region. The immature fruit is green with dense grey-white pubescence, most of which rubs off before maturity in late autumn when the fruit changes color to yellow with hard, strongly perfumed flesh.

Description of "Figure 22-5 Submenu with Three Radio Menu Items"

When the `DropShadow` effect is applied to the picture, it looks as shown in Figure 22-6.

Figure 22-6 Picture of Quince with a DropShadow Effect Applied



The quince is the sole member of the genus Cydonia and native to warm-temperate southwest Asia in the Caucasus region. The immature fruit is green with dense grey-white pubescence, most of which rubs off before maturity in late autumn when the fruit changes colour to yellow with hard, strongly perfumed flesh.

Description of "Figure 22-6 Picture of Quince with a DropShadow Effect Applied"

You can use the `setDisable` method of the `MenuItem` class to disable the No

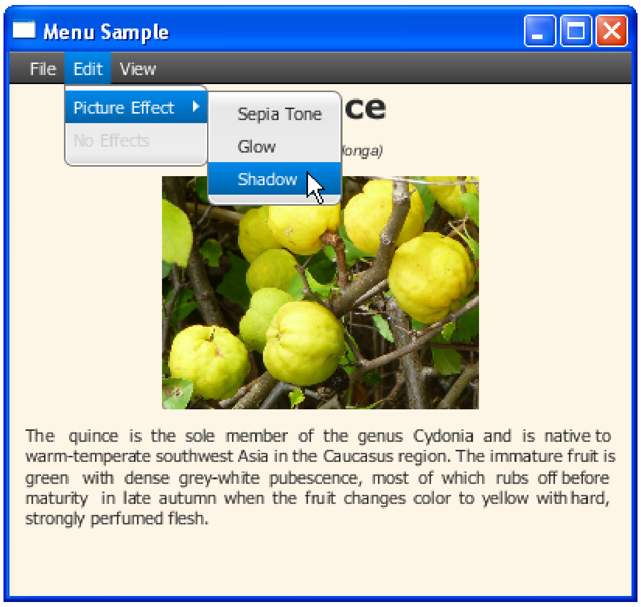
Effects menu when none of the effects are selected in the Picture Effect submenu. Modify Example 22-7 as shown in Example 22-8.

Example 22-8 Disabling a Menu Item

```
Menu menuEffect = new Menu("Picture Effect");
final ToggleGroup groupEffect = new ToggleGroup();
for (Entry<String, Effect> effect : effects) {
     RadioMenuItem itemEffect = new RadioMenuItem(effect.getKey());
     itemEffect.setUserData(effect.getValue());
     itemEffect.setToggleGroup(groupEffect);
     menuEffect.getItems().add(itemEffect);
}
final MenuItem noEffects = new MenuItem("No Effects");
noEffects.setDisable(true);
noEffects.setOnAction(new EventHandler<ActionEvent>() {
    public void handle(ActionEvent t) {
        pic.setEffect(null);
        groupEffect.getSelectedToggle().setSelected(false);
        noEffects.setDisable(true);
    }
});

groupEffect.selectedToggleProperty().addListener(new ChangeListener<Toggle>() {
    public void changed(ObservableValue<? extends Toggle> ov,
        Toggle old_toggle, Toggle new_toggle) {
            if (groupEffect.getSelectedToggle() != null) {
                Effect effect =
                    (Effect) groupEffect.getSelectedToggle().getUserData();
                pic.setEffect(effect);
                noEffects.setDisable(false);
            } else {
                noEffects.setDisable(true);
            }
        }
});
menuEdit.getItems().addAll(menuEffect, noEffects);
```

When none of the `RadioMenuItem` options are selected, the No Effect menu item is disabled as shown in Figure 22-7. When a user selects one of the visual effects, the No Effects menu item is enabled.

Figure 22-7 Effect Menu Item Is Disabled



The quince is the sole member of the genus Cydonia and is native to warm-temperate southwest Asia in the Caucasus region. The immature fruit is green with dense grey-white pubescence, most of which rubs off before maturity in late autumn when the fruit changes color to yellow with hard, strongly perfumed flesh.

Description of "Figure 22-7 Effect Menu Item Is Disabled"

Adding Context Menus

When you cannot allocate any space of your user interface for a required functionality, you can use a context menu. A context menu is a pop-up window that appears in response to a mouse click. A context menu can contain one or more menu items.

In the Menu Sample application, set a context menu for the picture of the plant, so that users can copy the image.

Use the `ContextMenu` class to define the context menu as shown in Example 22-9.

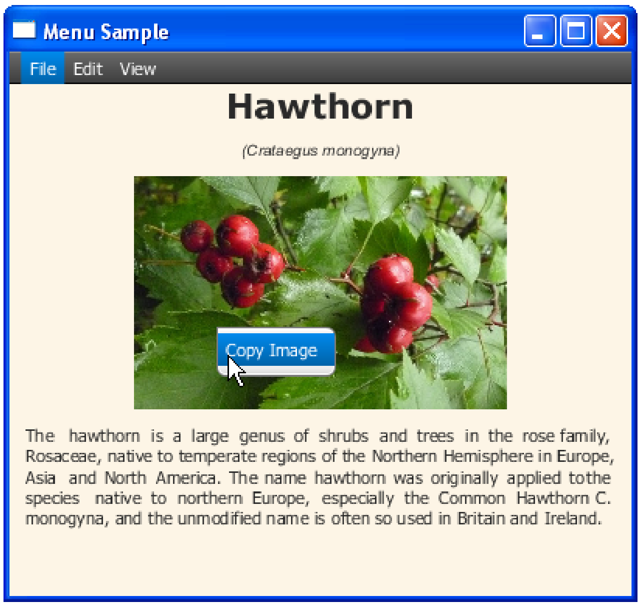
Example 22-9 Defining a Context Menu

```
final ContextMenu cm = new ContextMenu();
MenuItem cmItem1 = new MenuItem("Copy Image");
cmItem1.setOnAction(new EventHandler<ActionEvent>() {
    public void handle(ActionEvent e) {
        Clipboard clipboard = Clipboard.getSystemClipboard();
        ClipboardContent content = new ClipboardContent();
        content.putImage(pic.getImage());
        clipboard.setContent(content);
    }
});

cm.getItems().add(cmItem1);
pic.addEventHandler(MouseEvent.MOUSE_CLICKED,
    new EventHandler<MouseEvent>() {
        @Override public void handle(MouseEvent e) {
            if (e.getButton() == MouseButton.SECONDARY)
                cm.show(pic, e.getScreenX(), e.getScreenY());
        }
});
```

When a user right clicks the `ImageView` object, the `show` method is called for the context menu to enable its showing.

The `setOnAction` method defined for the Copy Image item of the context menu creates a `Clipboard` object and adds the image as its content. Figure 22-8 captures a moment when a user is selecting the Copy Image context menu item.

Figure 22-8 Using the Context Menu



Hawthorn

(Crataegus monogyna)

The hawthorn is a large genus of shrubs and trees in the rose family, Rosaceae, native to temperate regions of the Northern Hemisphere in Europe, Asia and North America. The name hawthorn was originally applied tothe species native to northern Europe, especially the Common Hawthorn C. monogyna, and the unmodified name is often so used in Britain and Ireland.

Description of "Figure 22-8 Using the Context Menu"

You can try to copy the image and paste it into in a graphical editor.

For further enhancements, you can add more menu items to the context menu and specify different actions. You can also create a custom menu by using the `CustomMenuItem` class. With this class you can embed an arbitrary node within a menu and specify, for example, a button or a slider as a menu item.

Related API Documentation

- `Menu`
- `MenuItem`
- `RadioMenuItem`
- `CheckMenuItem`
- `ContextMenu`
- `SeparatorMenuItem`
- `CustomMenuItem`

Previous Page Next Page