



# Принципы Java

9-1

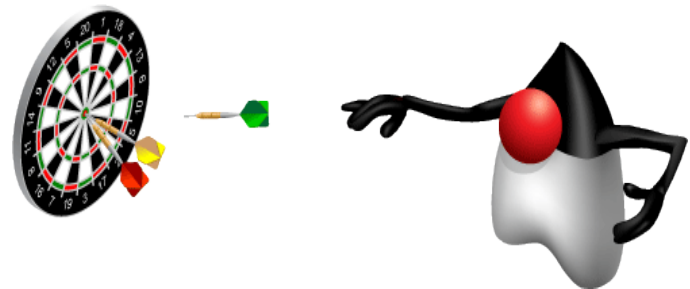
Общие сведения о JavaFX



# Цели

В данном уроке рассматриваются следующие темы:

- Создание проекта JavaFX
- Описать компоненты проекта JavaFX по умолчанию
- Описание различных типов узлов и панелей
- Описание графа сцены, корневого узла, сцен и этапов



# Темы

- Предварительный просмотр
- Создание программы JavaFX
- Корневой узел
- Граф сцены, сцена и этап



Общие сведения  
о JavaFX

Цвета и  
фигуры

Графика, звук  
и события  
мыши

Раздел 9

# Почти настало время итоговых экзаменов!

- Учиться очень важно.
- Любите ли вы учиться вместе с друзьями?
  - Но они живут в другом корпусе?
  - Где лучше всего встречаться с друзьями?
  - Что находится в самом центре кампуса?

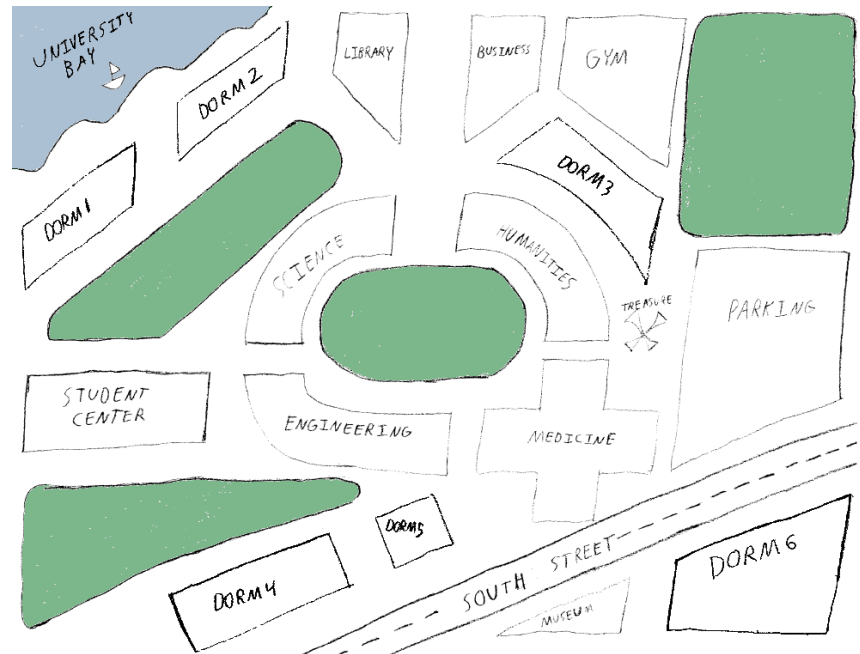
Спасибо, что напомнили...



# JavaFX может помочь

JavaFX используется для создания **приложений графического интерфейса пользователя**.

- GUI: Графический интерфейс пользователя
- Приложение GUI позволяет нам посмотреть ответ на карте.



- ORACLE®**  
Academy

# Но это не мой кампус!

- Вы правы.
- Было бы лучше, если в программе использовались следующие аспекты вашего образовательного учреждения...
  - Карта кампуса
  - Названия корпусов
  - Количество проживающих в корпусе
  - И ваша группа друзей
- Это набор задач данного раздела. В разделе 9 описывается все необходимое для повторного создания программы.



# Темы

- Предварительный просмотр
- Создание программы JavaFX
- Корневой узел
- Граф сцены, сцена и этап




Общие сведения  
о JavaFX

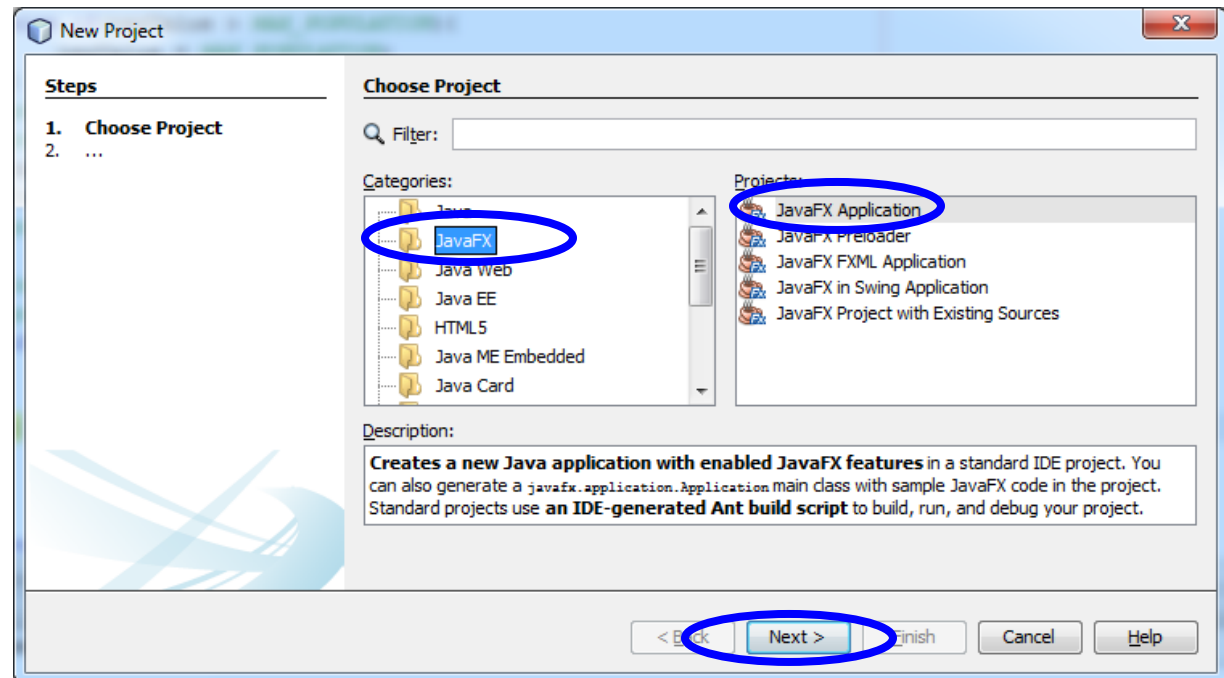
Цвета и  
фигуры

Графика, звук  
и события  
мыши

Раздел 9

# Как создать программу JavaFX

1. В NetBeans нажмите кнопку **New Project** (  )
2. Выберите **JavaFX** в поле "Category" и **JavaFX Application** в поле "Project", затем нажмите **Next**.
3. Продолжите создание так же, как и любой другой проект Java.





## Упражнение 2

- Создайте проект JavaFX.
  - Java должна предоставить программу по умолчанию.
- Поэкспериментируйте с программой. Можете ли вы внести данные изменения?
  - Измените метку кнопки.
  - Измените то, что выводится при нажатии кнопки.
  - Создайте другую кнопку и отобразите обе кнопки.
  - Измените стандартный размер окна приложения.

# Проект JavaFX по умолчанию

```
public class JavaFXTest extends Application {

    @Override
    public void start(Stage primaryStage) {
        Button btn = new Button();
        btn.setText("Сказать 'Hello World'");
        btn.setOnAction(new EventHandler<ActionEvent>() {

            @Override
            public void handle(ActionEvent event) {
                System.out.println("Hello World!");
            }
        });

        StackPane root = new StackPane();
        root.getChildren().add(btn);

        Scene scene = new Scene(root, 300, 250);

        primaryStage.setTitle("Hello World!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

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

# Два метода: `start()` и `main()`

- `start()` является входной точкой всех приложений JavaFX.

– Думайте о ней, как о главном методе для JavaFX.

```
public void start(Stage primaryStage) {  
    ...  
}
```

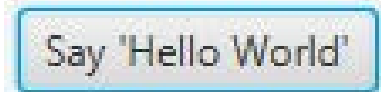
- Использование `main()` по-прежнему требуется в программах.

– Он запускает приложение JavaFX.

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

# Кнопки представляют собой объекты

- `Button` похож на любой другой объект.
  - Для них можно создать экземпляры.
  - Они содержат поля.
  - И методы.



```
public void start(Stage primaryStage) {  
    Button btn = new Button();  
    btn.setText("Сказать 'Hello World'");  
    ...  
}
```

- Из этого кода можно сказать, что ...
  - `Button` содержит поле текста.
  - `Button` содержит метод для изменения текстового поля.

# Кнопки представляют собой узлы

- Некоторые из этих полей и методов созданы для хранения и манипулирования **визуальными свойствами**:
  - `btn.getText()`
  - `btn.setMinHeight()`
  - `btn.setLayoutX()` //устанавливает  
положение x
  - `btn.setLayoutY()` //устанавливает  
положение y
  - `btn.isPressed()` //нажата ли кнопка?
- Объекты, которые похожи на этот, называются **узлами JavaFX**.

# Узлы

- Существует множество типов узлов JavaFX:



Кнопка



Прямоугольник



Круговая  
диаграмма



Полоса  
прокрутки



Текст



Графика

- Создаваемые визуальные объекты, скорее всего...
  - Будут узлом или
  - Будут включать узел в качестве поля



# Взаимодействие узлов

- Следующее помогает обрабатывать взаимодействие Button:

```
public void start(Stage primaryStage) {  
    ...  
    btn.setOnAction(new EventHandler<ActionEvent>() {  
        @Override  
        public void handle(ActionEvent event) {  
            System.out.println("Hello World!");  
        }  
    });  
    ...  
}
```

- Называется "анонимным внутренним классом".
  - Не выглядит ли синтаксис слишком беспорядочным?
  - **Лямбда-выражения** Java SE 8 представляют собой аккуратный альтернативный способ.
  - Обсуждение лямбда-выражений будет представлено далее в этом разделе.

# Темы

- Предварительный просмотр
- Создание программы JavaFX
- Корневой узел
- Граф сцены, сцена и этап



Раздел 9

# Создание узлов

- Экземпляры для узлов создаются так же, как для

```
public void start(Stage primaryStage) {  
    Button btn1 = new Button();  
    Button btn2 = new Button();  
    btn1.setText("Сказать 'Hello World'");  
    btn2.setText("222");  
    ...  
}
```

- После создания экземпляра узла:
  - Он существует и для хранения объекта выделяется объем памяти.
  - С его полями можно работать, а методы можно вызывать.
  - Но он может не отображаться...

← По крайней мере, пока...

# Отображение узлов

- Существует несколько этапов для отображения узла.

```
public void start(Stage primaryStage) {  
    Button btn1 = new Button();  
    Button btn2 = new Button();  
    btn.setText("Сказать 'Hello World'");  
    btn.setText("222");  
  
    StackPane root = new StackPane();  
    root.getChildren().add(btn1);  
    root.getChildren().add(btn2);  
    ...  
}
```

- Сначала, добавьте каждый узел в **корневой узел**.
  - Обычно он называется `root`.
  - Он очень похож на массив `ArrayList` для всех узлов.

# Добавление узлов в корневой узел

- Можно добавить каждый узел по отдельности.



```
root.getChildren().add(btn1);  
root.getChildren().add(btn2);  
root.getChildren().add(btn3);
```

- Или можно сразу добавить множество узлов.



```
root.getChildren().addAll(btn1, btn2, btn3);
```

- Но не добавляйте один и тот же узел несколько раз.  
– Это приводит к ошибке компилятора:



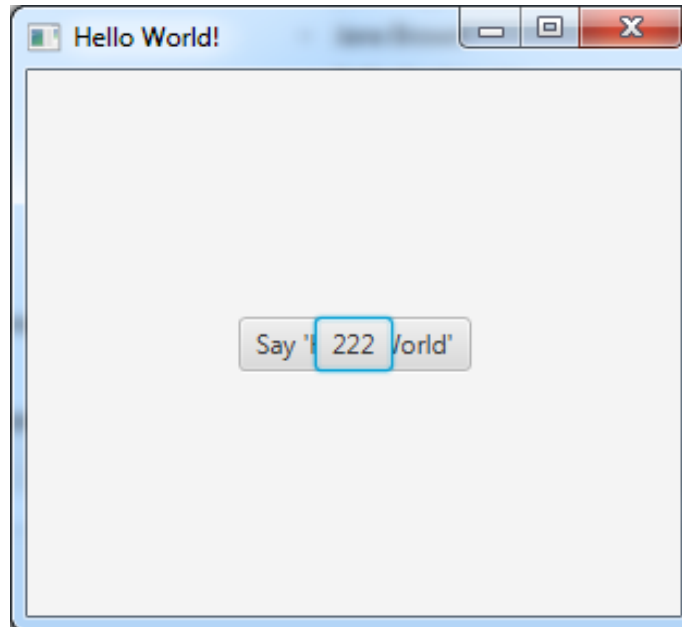
```
root.getChildren().add(btn1);  
root.getChildren().add(btn1);
```

# Корневой узел StackPane

- Корневым узлом в данном примере является StackPane.

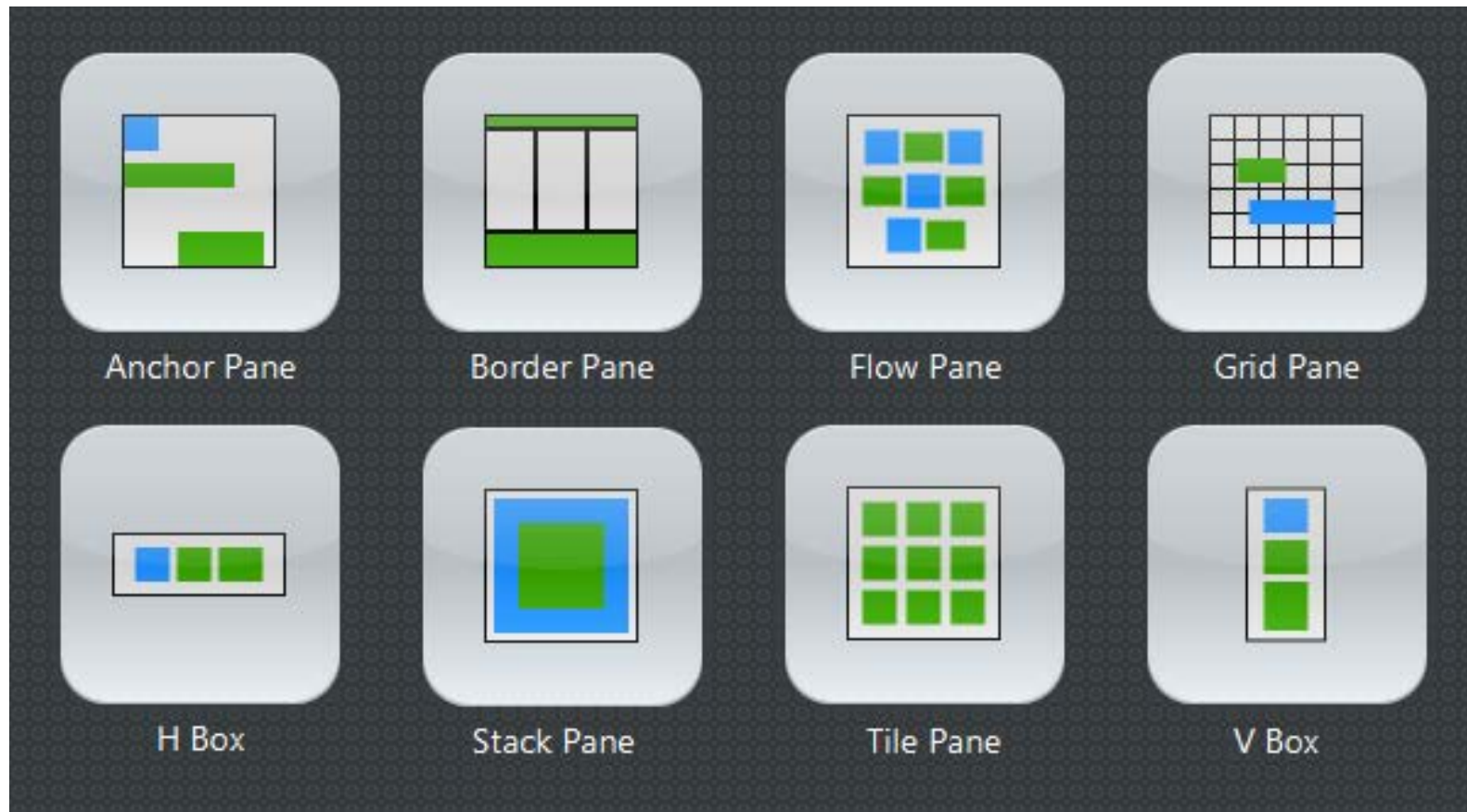
```
StackPane root = new StackPane();  
root.getChildren().addAll(btn1, btn2);
```

- StackPane создает стеки узлов поверх друг друга.
- Но маленькие кнопки могут затеряться или стать недоступными.



# Панели в качестве корневых узлов

Каждая **панель** определяет макет узлов.



# Программирование разных панелей в качестве корневых узлов

- Создать корневой узел в качестве отдельной панели легко.
- Просто укажите другой тип ссылки и тип объекта.

Измените это

И это

```
StackPane root = new StackPane();  
root.getChildren().addAll(btn1, btn2);
```

```
TilePane root = new TilePane();  
root.getChildren().addAll(btn1, btn2);
```

```
VBox root = new VBox();  
root.getChildren().addAll(btn1, btn2);
```





## Упражнение 3

- Измените текущий проект JavaFX.
  - Проведем небольшой эксперимент.
- После добавления кнопки в корневой узел попробуйте изменить расположение.
  - `btn1.setLayoutY(100);`
- Изменится ли расположение кнопки, если для корневого узла не было указано значение `StackPane`? Попробуйте выполнить следующие альтернативные способы:
  - `TilePane`
  - `VBox`
  - Группа

# Корневой узел группы

- Group позволяет размещать узлы в любом месте.

```
Group root = new Group();  
root.getChildren().addAll(btn1, btn2);  
btn1.setLayoutY(100);
```

- Панель может ограничивать возможность размещения узлов.
  - Их невозможно разместить даже при желании.
  - Невозможно нажать и перетащить узел, который заблокирован на панели.

```
StackPane root = new StackPane();  
root.getChildren().addAll(btn1, btn2);  
btn1.setLayoutY(100); //Не имеет эффекта
```

# Группа может содержать панель

- Панели также представляют собой узлы.
  - Любой узел можно добавить в корневой узел.
- Панель может быть отличным вариантом для хранения кнопок, диалоговых окон ввода текста и других элементов графического интерфейса.
  - Отдельные узлы на панели перемещать невозможно.
  - Но можно переместить всю панель в группу. Выполните ее перемещение, как и для любого другого узла.



## Упражнение 4

- Измените текущий проект JavaFX.
  - Можно еще поэкспериментировать.
- Можете ли вы сообразить, как сделать следующее?
  - Создать панель HBox и добавить в нее несколько кнопок.
  - Добавить панель HBox в корневой узел Group.
  - Расположить HBox ближе к нижней части окна.

# Темы

- Предварительный просмотр
- Создание программы JavaFX
- Корневой узел
- Граф сцены, сцена и этап



Общие сведения  
о JavaFX

Цвета и  
фигуры

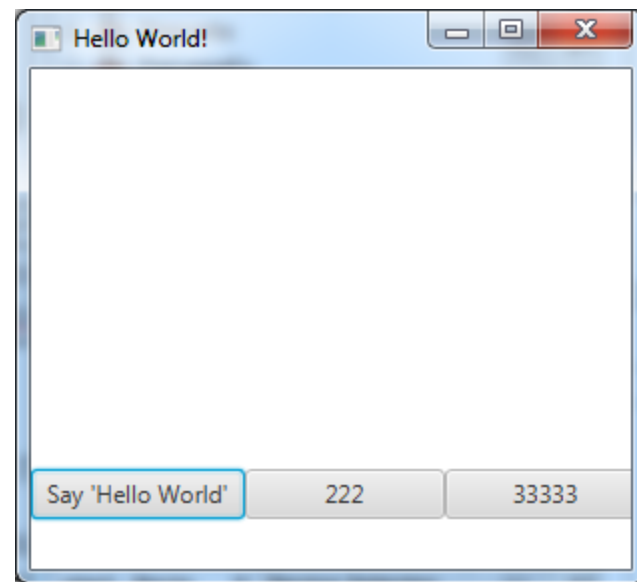
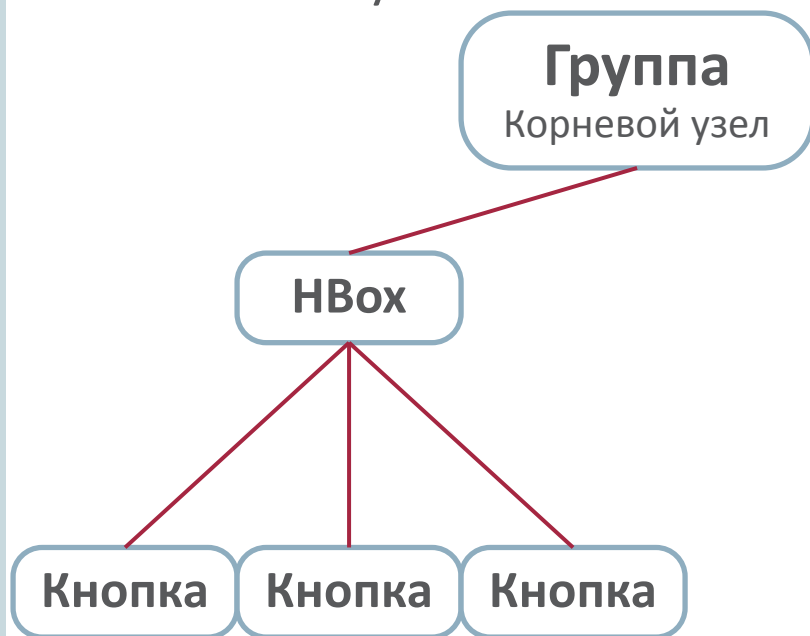
Графика, звук  
и события  
мыши

Раздел 9

# Граф сцены JavaFX

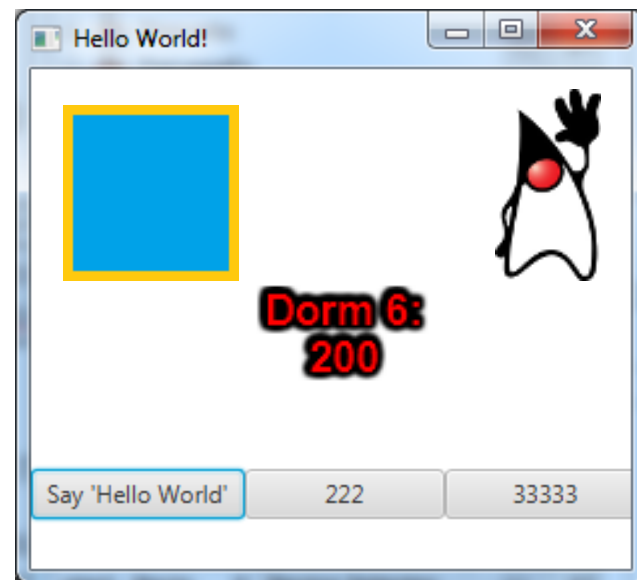
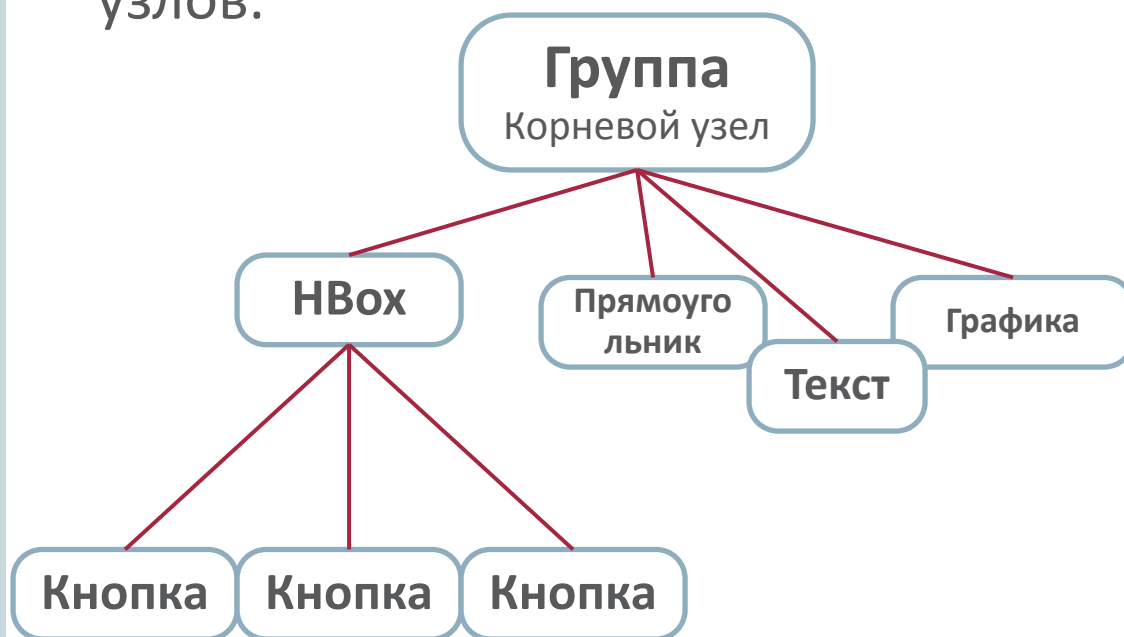
Способ добавления узлов, принятый вами, можно изобразить как **граф сцены**.

- Корневой узел содержит HBox.
- HBox выступает как контейнер для кнопок.



# Граф сцены

- HBox поддерживает графический интерфейс пользователя упорядоченным и удобно расположенным.
- Оставшаяся часть окна может быть использована для других узлов.



# Сцена и этап

Если посмотреть на оставшуюся часть программы JavaFX по умолчанию, мы заметим два дополнительных аспекта:

- Сцена (содержит корневой узел)
- Этап (содержит сцену)

```
public void start(Stage primaryStage) {  
    ...  
    Scene scene = new Scene(root, 300, 250);  
  
    primaryStage.setTitle("Hello World!");  
    primaryStage.setScene(scene);  
    primaryStage.show();  
}
```



# Что такое сцена?

Существует два важных свойства, которые описывают Scene:

- Граф сцены
  - Scene представляет собой контейнер для всего содержимого в графе сцены JavaFX.
- Размер
  - Для Scene можно установить ширину и высоту.
- Фон
  - В качестве фона можно установить значение Color или BackgroundImage.
- Сведения о курсоре
  - Scene может обнаруживать события движений мыши и обрабатывает свойства курсора.

```
Scene scene = new Scene(root, 300, 250, Color.BLACK);
```

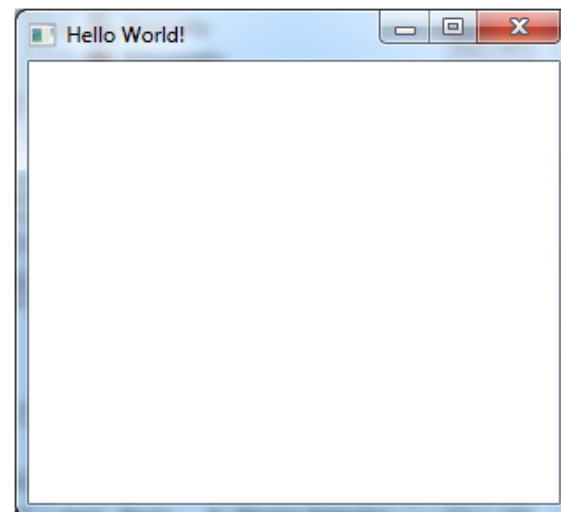
Корневой узел    ширина    рост    фон

# Что такое этап?

Рассматривайте этап как окно приложения.

Далее представлены два важных свойства этапа:

- Заголовок
  - Можно установить заголовок этапа.
- Сцена
  - Этап может содержать контейнер Scene.



```
primaryStage.setTitle("Hello World!");  
primaryStage.setScene(scene);  
primaryStage.show();
```

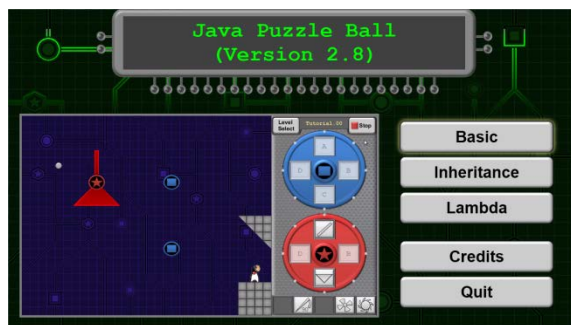
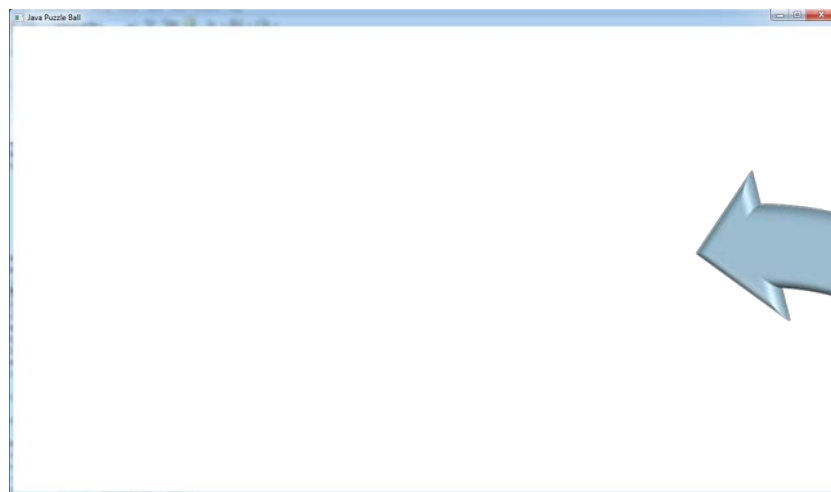
# Анимация иерархии

- Этап представляет собой контейнер верхнего уровня.
- Этап содержит сцену.
- Сцена содержит корневой узел.
- Корневой узел содержит другие узлы.



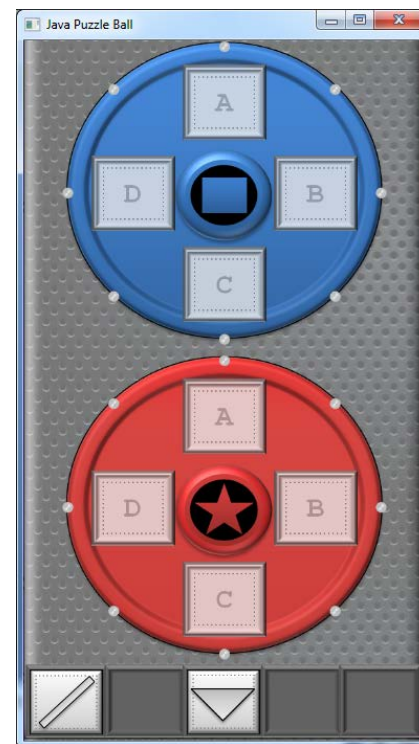
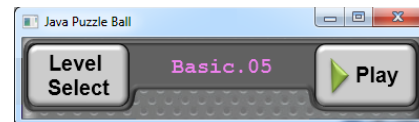
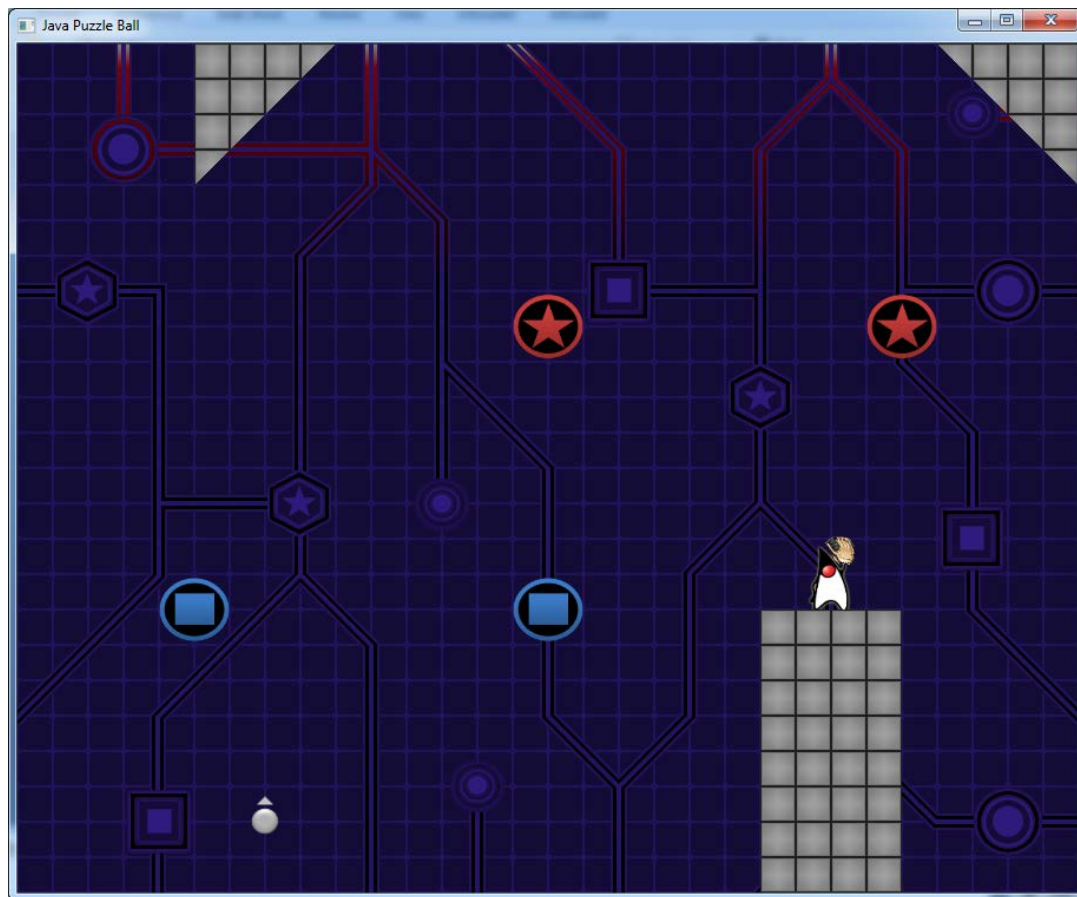
# Множество сцен, один этап

Можно поменять любую сцену на один этап.



# Множество сцен, множество этапов

Также можно создать множество этапов.



# Сводка

В данном уроке рассматриваются следующие темы:

- Создание проекта JavaFX
- Описать компоненты проекта JavaFX по умолчанию
- Описание различных типов узлов и панелей
- Описание графа сцены, корневого узла, сцен и этапов

