

Диалоговые окна

- основной элемент взаимодействия приложения с пользователем.
- единственное рекомендованное средство для программы, чтобы получить данные от человека

Общие требования

- окно должно быть как можно проще, не иметь слишком много кнопок
- следует максимально использовать стандартные, хорошо всем знакомые элементы
- содержимое диалогового окна нельзя делать прокручиваемым, если элементов много, то их делят на группы и размещают с помощью закладок или разделителей
- в диалоговые окна не добавляют меню

Виды диалоговых окон

Модальность

- не модальные (`show()`)
- модальные (`showAndWait()`)

Разработка

- стандартные
- пользовательские

Базовый класс Dialog<R>

R – результат работы диалога

Стандартный результат работы ButtonType

Преобразование результата

```
void setResultConverter(Callback<ButtonType,R>  
    value)
```

Получение результата работы

```
Optional<R> showAndWait()
```

Обработка результатов диалога

```
Optional<ButtonType> result = dialog.showAndWait();  
if (result.isPresent() && result.get() == ButtonType.OK) {  
    //do something  
}
```

```
dialog.showAndWait().ifPresent(response -> {  
    if (response == ButtonType.OK) { //do something  
    }  
});
```

Класс ButtonType

Специальный класс для задания кнопок в диалогах с пользователем

```
public static final ButtonType OK
```

```
public static final ButtonType CANCEL
```

```
ButtonType(String text, ButtonBar.ButtonData  
buttonData)
```

Класс Optional<T>

Контейнер для объекта, который получен
(существует) или нет

boolean isPresent()

T get()

void ifPresent(Consumer<? super T> consumer)

Optional<T> filter(Predicate<? super T>
predicate)

Создание пользовательского диалога

Создание объекта

```
Dialog<MyClass> dialog = new Dialog<>();
```

Создание внешнего вида диалога

```
dialog.getDialogPane().setContent(pane);
```

Добавление кнопок

```
ButtonType buttonTypeOk = new ButtonType("Okey", ButtonData.OK_DONE);  
dialog.getDialogPane().getButtonTypes().add(buttonTypeOk);
```

Преобразование возвращаемого значения

```
dialog.setResultConverter(new Callback<ButtonType, MyClass>(){...}
```

Получение результата

```
Optional<MyClass> result = dialog.showAndWait();
```


Окно сообщений Alert

```
public class Alert extends Dialog<ButtonType>
```

Конструкторы

```
Alert(Alert.AlertType alertType)
```

```
Alert(Alert.AlertType alertType, String contentText,  
      ButtonType... buttons)
```

Типы сообщений

```
AlertType.INFORMATION
```

```
AlertType.WARNING
```

```
AlertType.CONFIRMATION
```

```
AlertType.ERROR
```

Ввод строки текста Text Input Dialog

```
public class TextInputDialog  
    extends Dialog<String>
```

Конструктор

```
TextInputDialog(String defaultValue)
```

Метод вызова диалога

```
Optional<String> showAndWait()
```

Диалог выбора ChoiceDialog

```
public class ChoiceDialog<T>
```

```
extends Dialog<T>
```

Конструктор

```
ChoiceDialog(T defaultChoice, Collection<T>  
choices)
```

Метод вызова диалога

```
Optional<T> showAndWait()
```

Выбор файла FileChooser

public final class FileChooser extends Object

Методы

FileChooser()

File showOpenDialog(Window ownerWindow)

File showSaveDialog(Window ownerWindow)

void setInitialDirectory(File value)

void setSelectedExtensionFilter(FileChooser.ExtensionFilter filter)

List<File> showOpenMultipleDialog(Window ownerWindow)