Отображение списка объектов в виде таблицы

Строка – информация об очередном объекте списка

Столбец – значение поля объекта

class TableView<S>

S – тип объектов списка

Таблица состоит из столбцов

Class TableColumn<S,T>

T – тип содержимого каждой ячейки столбца

Методы класса TableView<S>

Конструкторы

TableView()

TableView(ObservableList<S> items)

Методы

ObservableList<S> getItems()

void setItems(ObservableList<S> value)

ObservableList<TableColumn<S,?>> getColumns()

void setEditable(boolean value)

Определение выделенной строки

Метод

TableView.TableViewSelectionModel<S> getSelectionModel()

Возвращаемое значение

Class TableView.TableViewSelectionModel<S>

Методы

S getSelectedItem()

ObservableList<S> getSelectedItems()

ReadOnlyObjectProperty<T> selectedItemProperty()

Требования к классу S

Поля класса, отображаемые в таблице должны быть свойствами

SimpleStringProperty name;

StringProperty name;

Методы:

public String getName()

public void setName(String name)

Столбцы таблицы TableColumn<S,T>

Конструктор

TableColumn(String text)

Методы

void setMinWidth(double value)

void setMaxWidth(double value)

void setResizable(boolean value)

void setSortable(boolean value)

ObservableList<TableColumn<S,?>> getColumns()

Связывание столбца с полем класса S

void setCellValueFactory (Callback <TableColumn.CellDataFeatures<S,T>,ObservableValue<T>> value)

Интерфейс Interface Callback<P,R> определяет функцию преобразования типа P в тип R (R call(P param))

TableColumn.CellDataFeatures<S,T> класс-обертка для отдельной ячейки таблицы

ObservableValue<T> – интерфейс для наблюдаемого объекта

Класс реализующий интерфейс Callback <TableColumn.CellDataFeatures<S,T>, ObservableValue<T>>

class PropertyValueFactory<S,T>

PropertyValueFactory(String property)

Пример

```
TableColumn<Organization, String>
nameCol = new TableColumn<>("Name");
nameCol.setMinWidth(130);
nameCol.setCellValueFactory
            (new PropertyValueFactory("name"));
```

Преобразование ячейки таблицы в поле для редактирования

Метод определяющий построение ячеек таблицы

```
void setCellFactory(Callback<TableColumn<S,T>,
                            TableCell<S,T>> value)
```

Аргумент метода можно получить с помощью класса TextFieldTableCell<S,T> и его статических методов

```
Callback<TableColumn<S,String>,TableCell<S,String>>
                    forTableColumn()
```

```
Callback<TableColumn<S,T>,TableCell<S,T>>
    forTableColumn(StringConverter<T> converter)
```

Пример

```
nameCol.setCellValueFactory(
        new PropertyValueFactory<Organization,
                                 String>("name"));

nameCol.setCellFactory(
        TextFieldTableCell.forTableColumn());

personnelCol.setCellValueFactory(
        new PropertyValueFactory<Organization,
                                 Integer>("personnel"));

personnelCol.setCellFactory(
      TextFieldTableCell.forTableColumn(
                        new IntegerStringConverter()));
```

Обработка события редактирования ячейки

void setOnEditCommit (EventHandler <TableColumn.CellEditEvent<S,T>> value)

Событие CellEditEvent<S,T>

TablePosition<S,T> getTablePosition()

TableView<S> getTableView()

T getNewValue()

T getOldValue()

Определение строки и столбца таблицы

value.getTablePosition().getRow()

value.getTablePosition().getColumn()

Пример

```
nameCol.setOnEditCommit(
  new EventHandler<CellEditEvent<Organization, String>>() {
     @Override
     public void handle(CellEditEvent<Organization, String> t) {
        ((Organization) t.getTableView().getItems().get(
              t.getTablePosition().getRow())).
              setName(t.getNewValue());
        }
   }
  );
```