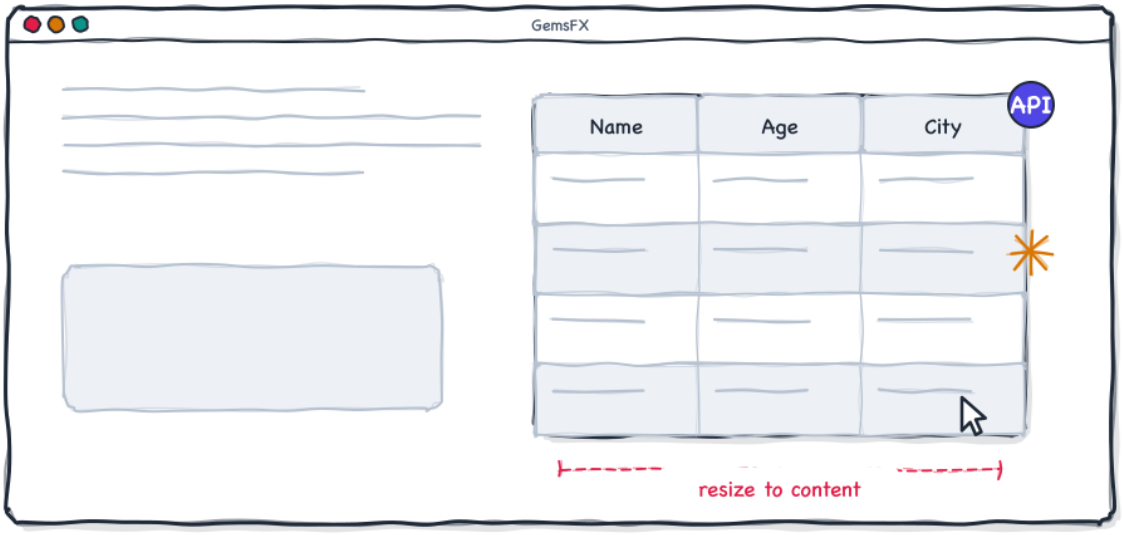
AdvancedTableView

Developer Manual

GemsFX • com.dlsc.gemsfx • JavaFX Custom Controls



Generated cartoon overview of AdvancedTableView.

AdvancedTableView is a small extension of JavaFX TableView that installs a custom header stack and exposes an API for resizing all columns to fit their content.

Table of Contents

1. Introduction

AdvancedTableView extends `TableView`. It keeps the standard JavaFX table API and skin behaviour, but replaces the header row classes so every leaf header is an `AdvancedTableColumnHeader`. The only public GemsFX addition is `autoResizeAllColumns(...)`.

1.1 Key features

- Constructors mirror TableView: empty or with an ObservableList of items.
- Uses AdvancedTableViewSkin, AdvancedTableHeaderRow and AdvancedNestedTableColumnHeader.
- autoResizeAllColumns() measures the first 100 rows.
- autoResizeAllColumns(int rows) ignores non-positive row counts.
- If called before a skin exists, the resize request is deferred until skin creation.

1.2 Maven dependency

```
<dependency>
    <groupId>com.dlsc.gemsfx</groupId>
    <artifactId>gemsfx</artifactId>
    <version>4.4.1</version>
</dependency>
```

Use package `com.dlsc.gemsfx`.

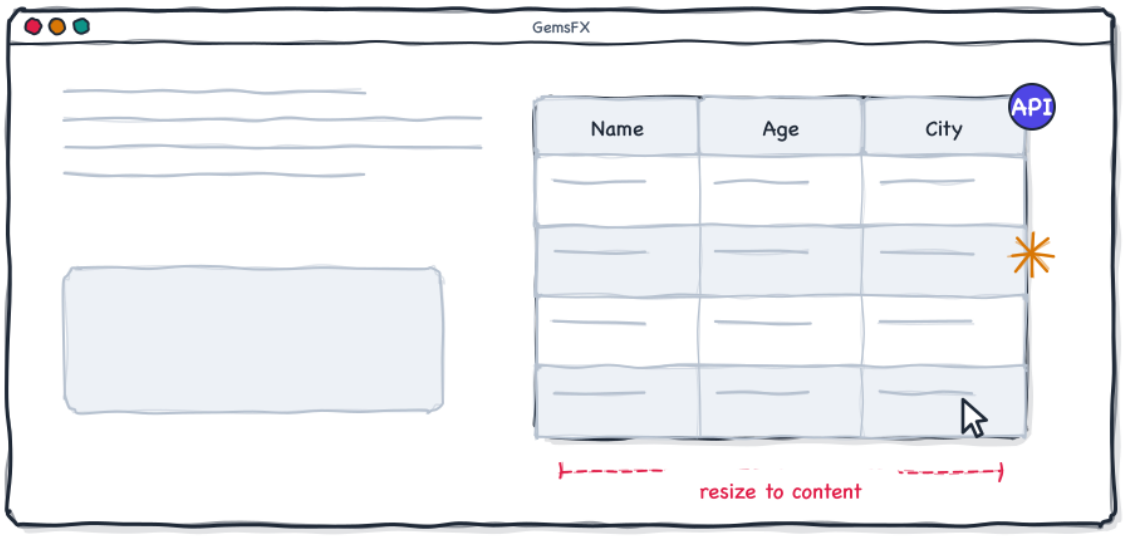
2. Getting started

The snippet below uses only APIs verified in the source and demo code.

```
AdvancedTableView<Person> table = new AdvancedTableView<>();
TableColumn<Person, String> first = new TableColumn<>("First");
first.setCellValueFactory(new PropertyValueFactory<>("firstName"));
TableColumn<Person, String> last = new TableColumn<>("Last");
last.setCellValueFactory(new PropertyValueFactory<>("lastName"));
table.getColumns().setAll(first, last);
table.getItems().setAll(people);

// after items / columns are available
table.autoResizeAllColumns();
```

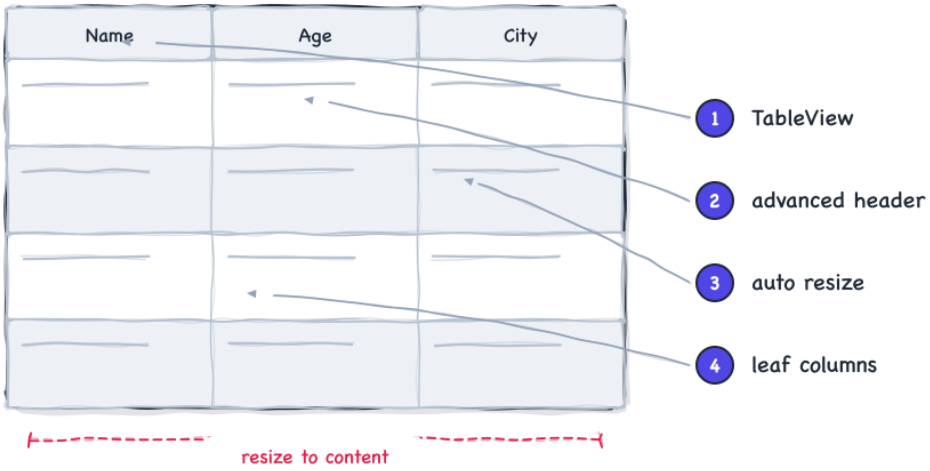
Minimal setup for `AdvancedTableView`.



A first look at the control.

3. Anatomy

The diagram and table identify the nodes, model objects and style classes that matter when using or styling the control.



The main parts of the control.

Part	Type / style	Description
AdvancedTableView	TableView subclass	Keeps the standard TableView item, column, selection and sorting APIs.
AdvancedTableViewSkin	TableViewSkin	Installs the advanced table header row.
AdvancedTableHeaderRow	TableHeaderRow	Creates the advanced root nested header.

Part	Type / style	Description
AdvancedNestedTableColumnHeader	NestedTableColumnHeader	Creates nested or leaf advanced column headers.
AdvancedTableColumnHeader	TableColumnHeader	Exposes resizeColumnToFitContent for each leaf column.

4. Control API

4.1 Constructors and methods

Property	Type	Default	Description
AdvancedTableView()	constructor	empty TableView	Creates an empty table and installs deferred auto-resize support.
AdvancedTableView(ObservableList<T>)	constructor	items from argument	Creates a table with initial items.
autoResizeAllColumns()	void	100 rows	Resizes all leaf columns by measuring content in the first 100 rows.
autoResizeAllColumns(int rows)	void	caller supplied	Resizes all leaf columns using the given row limit; rows <= 0 do nothing.

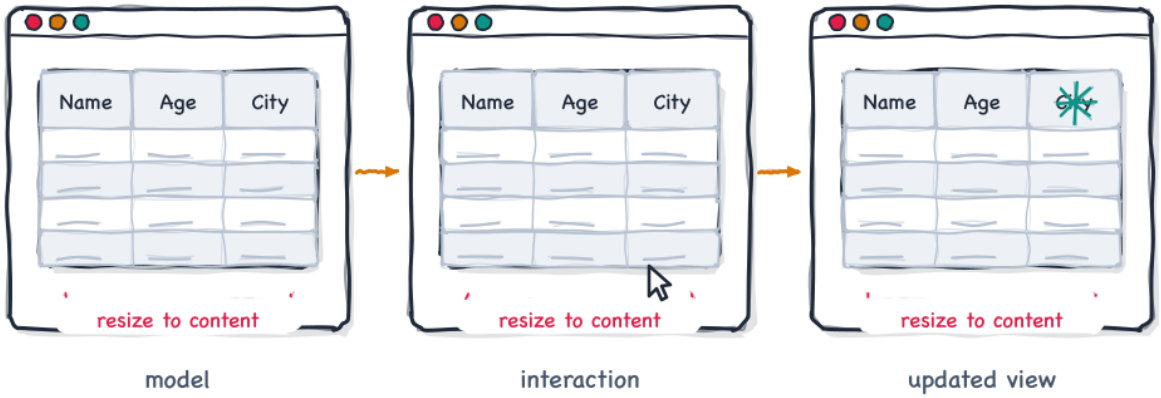
4.2 Deferred state

Property	Type	Default	Description
autoResizeAllColumns	boolean field	false	Internal flag storing whether a resize call happened before skin creation.
autoResizeRows	int field	0	Internal row count used by the deferred resize path.

5. Behaviour

5.1 Column auto-sizing

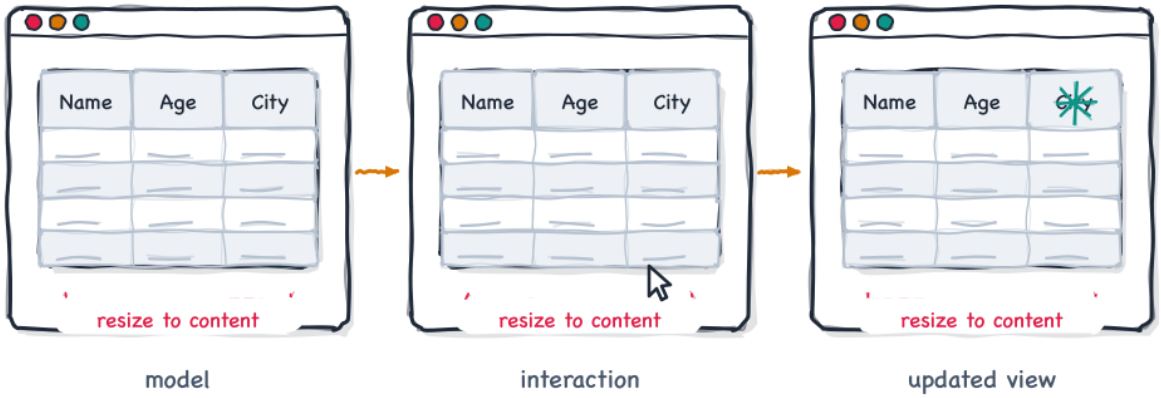
The resize method retrieves the skin root header and recursively walks nested headers. Leaf AdvancedTableColumnHeader instances delegate to JavaFX resizeColumnToFitContent(rows).



The main runtime behaviour.

5.2 Skin timing

If the skin is null, the control records that all columns should be resized later. A listener on skinProperty performs the resize once a skin is available.



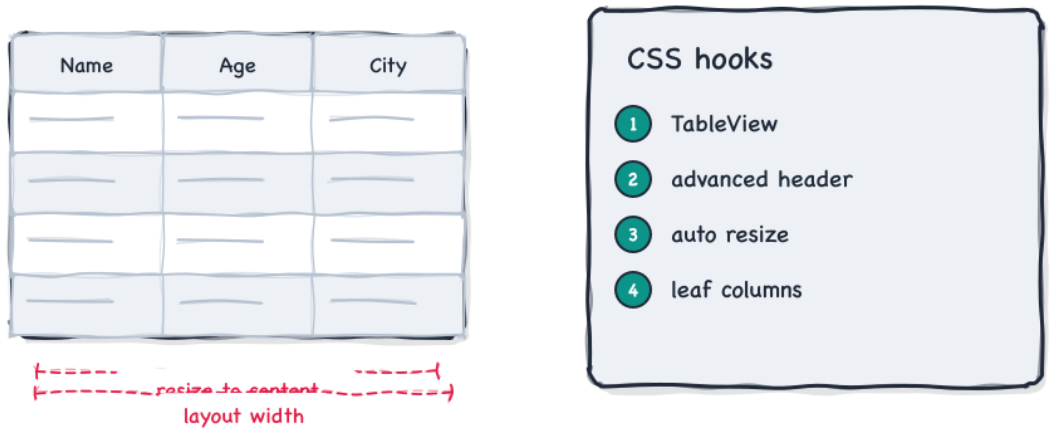
Data and interaction flow.

5.3 Standard TableView behaviour

Sorting, selection, columns, row factories and cell factories remain the standard JavaFX TableView APIs; GemsFX does not replace them.

6. Styling

The style hooks below were verified in the control, skin and CSS sources.



Style hooks and visual states.

6.1 Style classes

Style class	Where used
none	No dedicated GemsFX stylesheet or style classes were found.

6.2 Pseudo classes

Pseudo class	Meaning
none	No pseudo classes are set by this control.

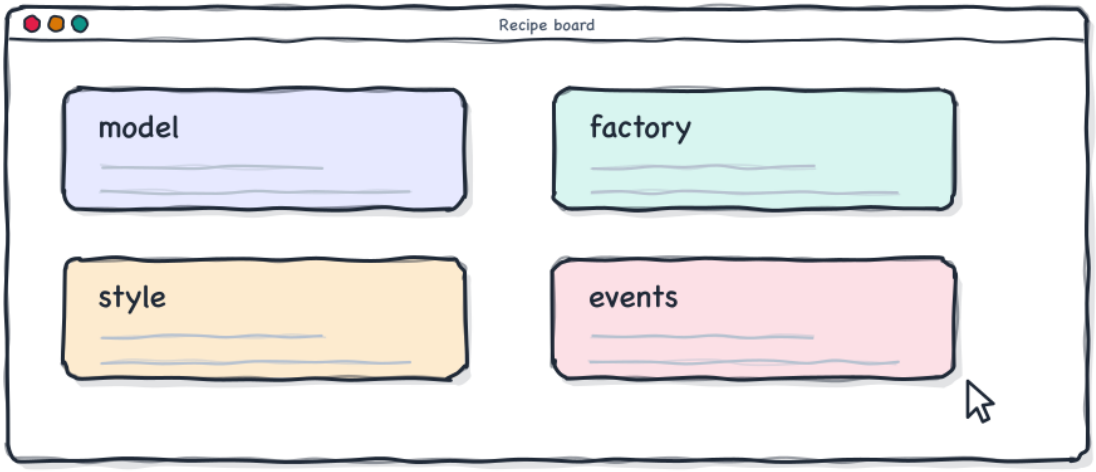
6.3 Styleable CSS properties

Property	Type	Default	Description
none			No styleable CSS properties are declared by this control.

```
/* AdvancedTableView uses the standard JavaFX TableView stylesheet hooks. */
.advanced-table-view .column-header {
    -fx-font-weight: bold;
}
```

Example CSS.

7. Recipes



Common configuration recipes.

7.1 Resize after loading data

```
table.getItems().setAll(loadPeople());
table.autoResizeAllColumns(200);
```

7.2 Resize after adding columns

```
table.getColumns().setAll(firstNameColumn, lastNameColumn, emailColumn);
table.autoResizeAllColumns();
```

7.3 Ignore tiny samples

```
// rows <= 0 is intentionally ignored
table.autoResizeAllColumns(0);
```

7.4 Use normal TableView APIs

```
table.getSelectionModel().selectedItemProperty().addListener((obs, old, person) -> {
    System.out.println(person);
});
```

7.5 Checklist

1. Populate columns before calling autoResizeAllColumns.
2. Pass a row count large enough to represent typical content.
3. Use ordinary TableView CSS and APIs for everything except auto-resizing.
4. There is no GemsFX-specific CSS file, resource bundle or accessibility setup.

8. See also

No dedicated demo app was found in `gemsfx-demo` for this control.

- Related GemsFX controls: GridTableView, TableView, MultiColumnListView.
- API documentation: https://dlsc-software-consulting-gmbh.github.io/GemsFX/api/