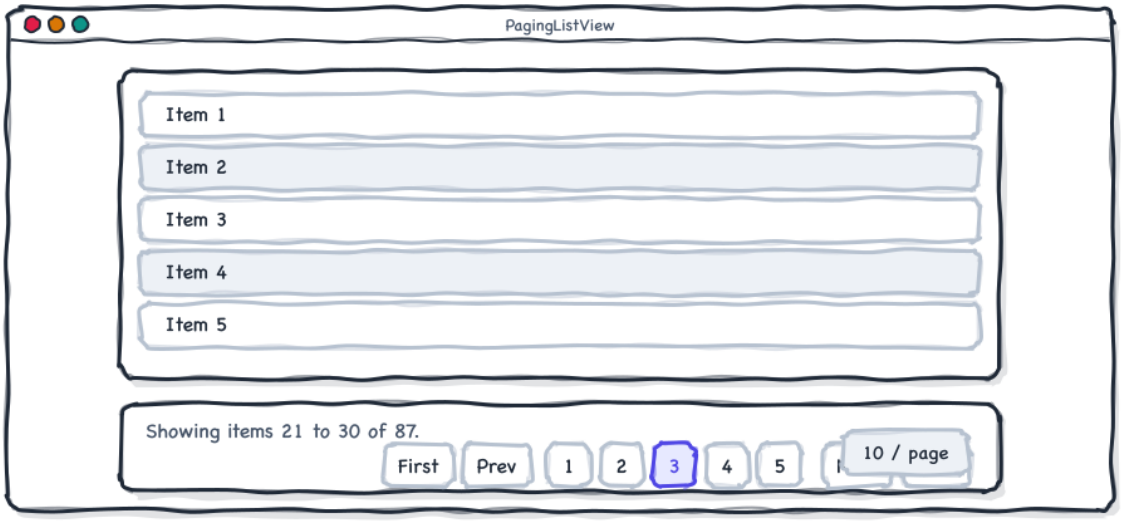
PagingListView

Developer Manual

GemsFX • com.dlsc.gemsfx.paging • JavaFX Custom Controls



A ListView page and its controls are loaded through the shared paging callback contract.

PagingListView combines a ListView, PagingControls and the ItemPagingControlBase loading contract. It requests one page at a time from a callback, displays the returned items and exposes loading, placeholder, selection and open-item hooks.

Table of Contents

1. Introduction 2

1.1 Key features 2

1.2 Maven dependency 2

2. Getting started 2

3. Anatomy 3

4. Control API 4

4.1 Shared item paging properties 4

4.2 List-specific properties 4

5. Loader contract 5

6. Loading, empty and error states 5

7. Layout and refresh 6

8. Styling 7

9. Localization 7

10. Accessibility 8

11. Recipes 8

11.1 Use an in-memory list 8

11.2 Open selected item 8

11.3 Hide controls for a fixed first page 8

11.4 Refresh cells without loading 8

11.5 Checklist 8

12. See also 8

1. Introduction

PagingListView is a paged list control. It owns an internal `ListView`, binds that list to the current page items, and uses an internal `LoadingService` to call the configured loader.

1.1 Key features

- Asynchronous loader based on PagingLoadRequest and PagingLoadResponse.
- Embedded PagingControls with top or bottom placement.
- Multiple-selection model from the internal ListView.
- Custom cell factory and progress indicator.
- Placeholder for empty pages.
- reload() restarts the service; refresh() refreshes visible cells.

1.2 Maven dependency

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

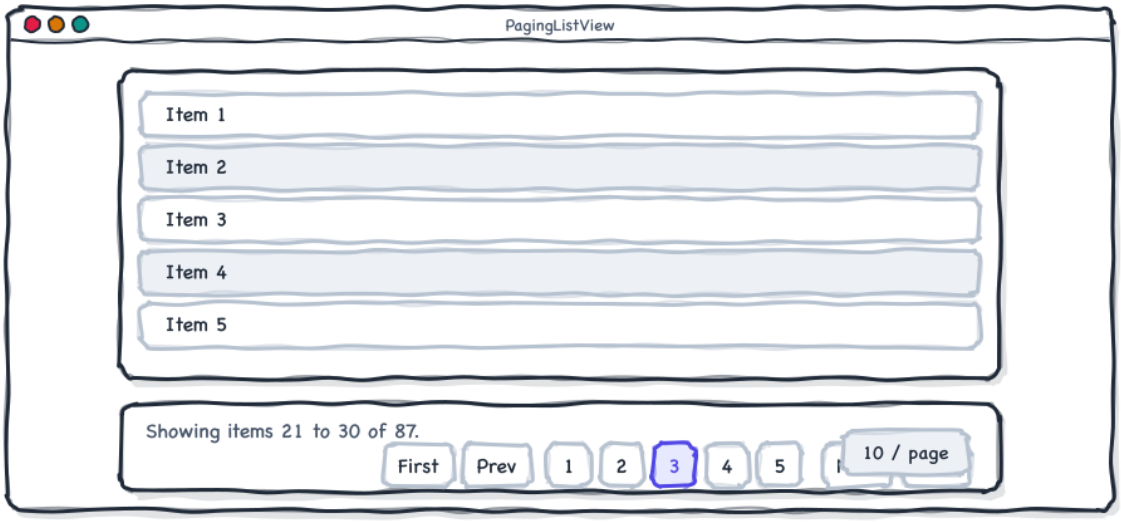
The class is in `com.dlsc.gemsfx.paging`.

2. Getting started

```
PagingListView<Person> view = new PagingListView<>();
view.setLoader(request -> {
    int offset = request.getPage() * request.getPageSize();
    List<Person> page = repository.find(offset, request.getPageSize());
    return new PagingLoadResponse<>(page, repository.count());
});

view.setCellFactory(list -> new ListCell<>() {
    @Override
    protected void updateItem(Person item, boolean empty) {
        super.updateItem(item, empty);
        setText(empty || item == null ? null : item.getLastName());
    }
});
```

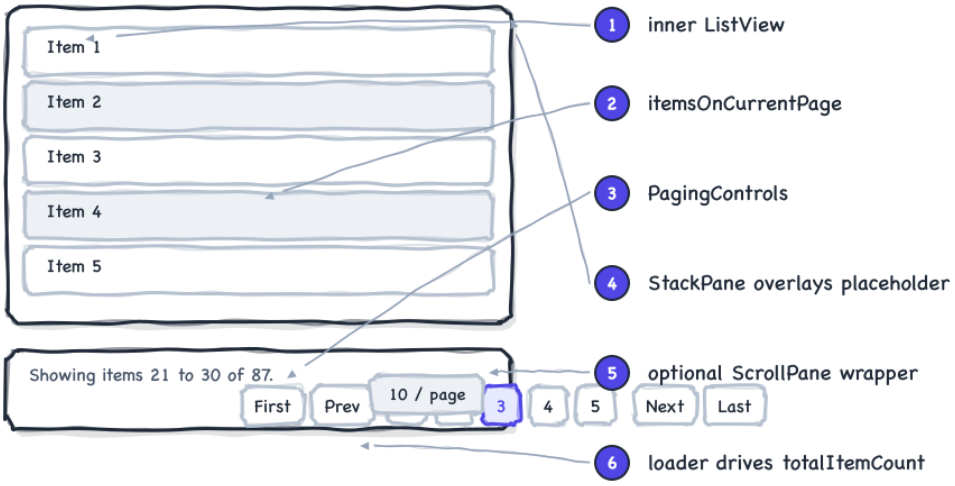
A complete loader and cell factory.



PagingListView wraps a ListView and a PagingControls bar.

3. Anatomy

The skin places the internal ListView in a StackPane so it can overlay a placeholder or loading indicator. PagingControls are placed above or below that content.



The parts of PagingListView.

Part	Style class	Description
.paging-list-view	Root style class.	
.inner-list-view	Internal ListView.	
.stack-pane	Stack containing content, placeholder and progress.	

Part	Style class	Description
.content	Container spacing for view and paging controls.	
.list-cell:selected	Selected cell styling from paging-list-view.css.	

4. Control API

4.1 Shared item paging properties

Property	Type	Default	Description
loader	ObjectProperty<Callback<PagingLoadRequest, PagingLoadResponse<T>>>	LoadingService default callback	Callback that receives zero-based page and pageSize and returns items plus the total count.
itemsOnCurrentPage	ReadOnlyListProperty<T>	empty list	Items returned by the latest successful load and shown by the inner view.
loadingStatus	ReadOnlyObjectProperty<LoadingStatus>	OK	OK, LOADING or ERROR based on the internal LoadingService state.
loadDelayInMillis	LongProperty	50	Delay before the control starts a load. Negative values are rejected.
commitLoadStatusDelay	LongProperty	400	Delay before the loading status is committed to the UI.
fillLastPage	BooleanProperty	false	Pads the last page layout where supported. Styleable as -fx-fill-last-page.
pagingControlsLocation	ObjectProperty<Side>	Side.BOTTOM	Places controls at TOP or BOTTOM. LEFT and RIGHT are not supported and reset to BOTTOM.
showPagingControls	BooleanProperty	true	Shows or hides embedded PagingControls. Styleable as -fx-show-paging-controls.
usingScrollPane	BooleanProperty	false	Wraps the content in a ScrollPane when true. Styleable as -fx-use-scroll-pane.
placeholder	ObjectProperty<Node>	Label("No items")	Node shown when the current page has no content.
onOpenItem	ObjectProperty<Consumer<T>>	null	Callback invoked by the inner item view when an item is opened.
selectionModel	ObjectProperty<MultipleSelectionModel<T>>	control-specific model	Selection model of the inner list or grid table.

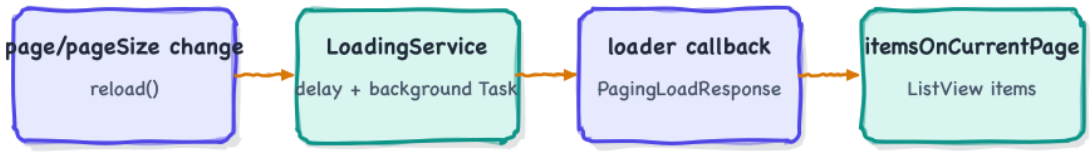
4.2 List-specific properties

Property	Type	Default	Description
cellFactory	ObjectProperty<Callback<ListView<T>, ListCell<T>>>	default text cell	Factory used by the internal ListView.

Property	Type	Default	Description
progressIndicator	ObjectProperty<Node>	CircleProgressIndicator	Node shown while loading is committed.

5. Loader contract

The loader receives a `PagingLoadRequest` with zero-based page and pageSize. It must return a non-null item list and the total item count for the whole data set. The control updates `itemsOnCurrentPage` and page count from the response.



Data flow from page change to ListView items.

> **Tip.** For in-memory lists, `SimpleLoader` slices the backing list with offset `page * pageSize` and reloads when that list changes.

6. Loading, empty and error states

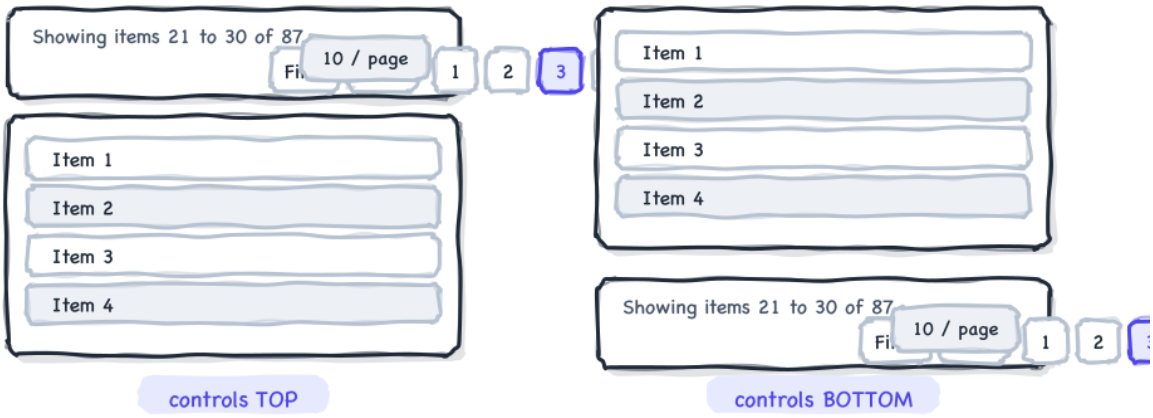
While the service is running, loadingStatus becomes LOADING. On success it becomes OK; on failure it becomes ERROR. The placeholder is visible when no items are available for the current page.

Normal, loading and placeholder states.

- loadDelayInMillis defaults to 50 ms on the control.
- commitLoadStatusDelay defaults to 400 ms.
- Negative load delays are rejected.
- Cancelled service work is ignored.

7. Layout and refresh

Paging controls can appear above or below the list. LEFT and RIGHT locations are not supported by ItemPagingControlBase and are reset to BOTTOM.



Top and bottom paging-control placement.

`reload()` starts a new load from the configured loader. `refresh()` only refreshes the visible ListView cells and does not query the data source.

8. Styling

Style class	Description
.paging-list-view	Root style class.
.inner-list-view	Internal ListView.
.stack-pane	Stack containing content, placeholder and progress.
.content	Container spacing for view and paging controls.
.list-cell:selected	Selected cell styling from paging-list-view.css.

CSS property	Type	Default
-fx-fill-last-page	boolean	false
-fx-paging-controls-location	Side	BOTTOM
-fx-show-paging-controls	boolean	true
-fx-use-scroll-pane	boolean	false
-fx-show-page-size-selector	boolean	true
-fx-same-width-page-buttons	boolean	false
-fx-show-previous-next-page-button	boolean	true
-fx-message-label-strategy	enum	ALWAYS_SHOW
-fx-first-last-page-display-mode	enum	SHOW_PAGE_BUTTONS
-fx-max-page-indicators-count	integer	5
-fx-page-alignment	HPos	RIGHT

```
.paging-list-view {
    -fx-paging-controls-location: TOP;
    -fx-show-paging-controls: true;
}
.paging-list-view .inner-list-view .list-cell:selected {
    -fx-font-weight: bold;
}
```

Example CSS for PagingListView.

9. Localization

The embedded PagingControls use the paging-control bundle. The default placeholder comes from the item-paging-control bundle and displays *No items*.

10. Accessibility

PagingListView sets `AccessibleRole.LIST_VIEW`. The embedded list exposes normal ListView selection and focus behaviour.

11. Recipes

11.1 Use an in-memory list

```
ObservableList<Person> all = FXCollections.observableArrayList();
view.setLoader(new SimpleLoader<>(all));
```

11.2 Open selected item

```
view.setOnOpenItem(person -> showDetails(person));
```

11.3 Hide controls for a fixed first page

```
view.setShowPagingControls(false);
```

11.4 Refresh cells without loading

```
view.refresh();
```

11.5 Checklist

1. Return the total count for the entire data set, not just the page.
2. Keep loader work off the JavaFX application thread.
3. Use reload() after changing query parameters.
4. Use refresh() only when existing cell visuals changed.

12. See also

- Demo app: `PagingListViewApp`. Run with `mvn javafx:run -f gemsfx-demo/pom.xml -Dmain.class=com.dlsc.gemsfx.demo.PagingListViewApp`.
- Related controls: PagingControls and PagingGridTableView.
- API docs: https://dlsc-software-consulting-gmbh.github.io/GemsFX/api/