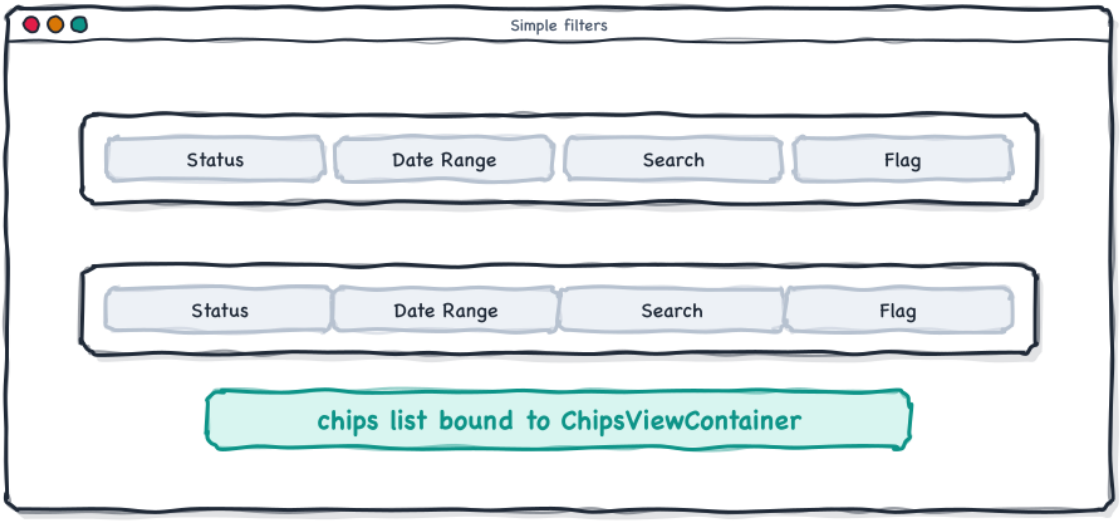
SimpleFilterView

Developer Manual

GemsFX • com.dlsc.gemsfx • JavaFX Custom Controls



SimpleFilterView arranges compact filter inputs and exposes their values as chips.

SimpleFilterView is a lightweight HBox helper that creates common filter input controls and mirrors their selected values as ChipView objects. It does not filter data itself; applications consume its controls and onChange callback.

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 Factory methods 4
5. Layout modes 4
6. Chips and clear behaviour 5
7. Using values in an application 6
8. Styling 6
9. Localization 7
10. Accessibility 7
11. Recipes 7
11.1 Use compact mode 7
11.2 Bind chips to a container 7
11.3 Single-selection enum 7
11.4 Reload after change 7
11.5 Checklist 7
12. See also 7

1. Introduction

SimpleFilterView extends `HBox`. It is not skinned and does not own a predicate pipeline. Instead, it provides factory methods for common input controls and a chips list suitable for `ChipsViewContainer`.

1.1 Key features

- Factory methods for SelectionBox, CheckBox, text fields, search fields, DateRangePicker, CalendarPicker and DatePicker.
- STANDARD and COMPACT layout modes.
- Automatic selection-item, first, middle, last and only style classes on child controls.
- Read-only chips list with close actions that clear the source control.
- clear() resets supported controls and invokes onChange once.

1.2 Maven dependency

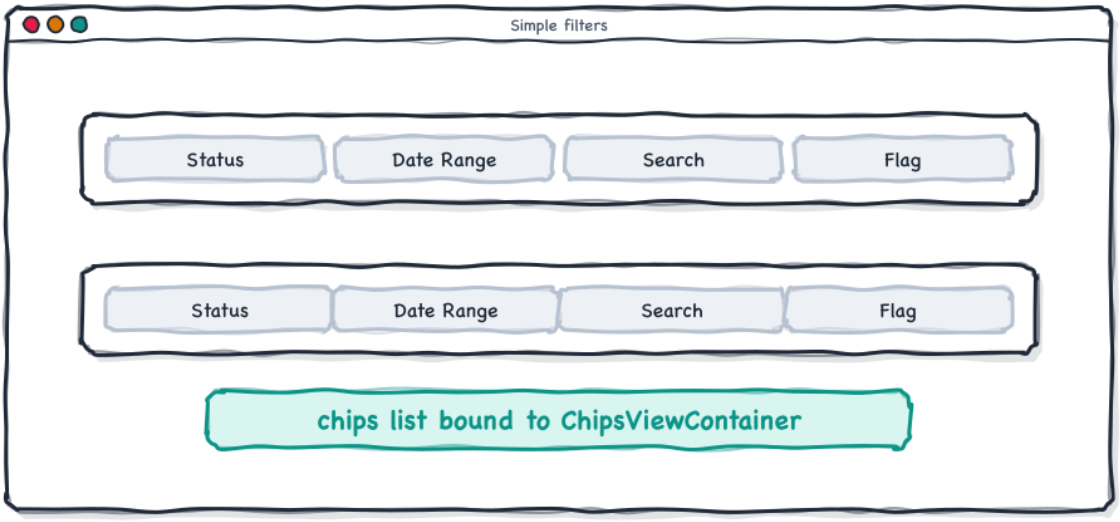
```
<dependency>
    <groupId>com.dlsc.gemsfx</groupId>
    <artifactId>gemsfx</artifactId>
    <version>4.4.1</version>
</dependency>
```

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

2. Getting started

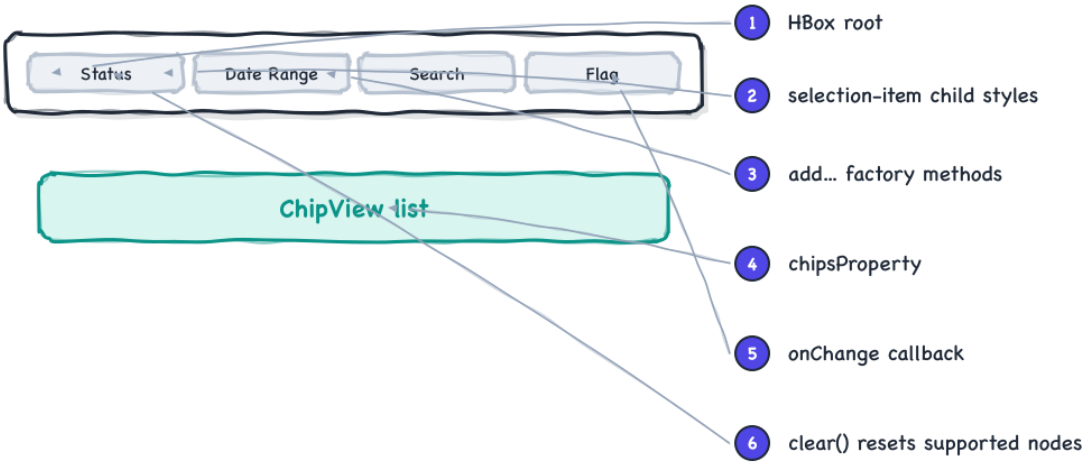
```
SimpleFilterView filters = new SimpleFilterView();
SelectionBox<Status> status = filters.addSelectionBox("Status", Status.class);
DateRangePicker range = filters.addDateRangePicker("Date Range");
CustomTextField search = filters.addSearchTextField("Search");

ChipsViewContainer chips = new ChipsViewContainer();
chips.chipsProperty().bind(filters.chipsProperty());
chips.setOnClear(filters::clear);

filters.setOnChange(() -> reload(status.getSelectionModel().getSelectedItems(),
                                 range.getValue(), search.getText()));
```

A SimpleFilterView with chips and a reload callback.



SimpleFilterView with standard controls, compact controls and bound chips.

3. Anatomy



The parts of SimpleFilterView.

Part	Description
HBox root	The control itself; style class simple-filter-view.
Factory-created children	Supported JavaFX and GemsFX inputs.
selection-item classes	Child style classes for first, middle, last or only.
chips property	Read-only chips generated from selected or entered values.
onChange	Application callback after value changes.

4. Control API

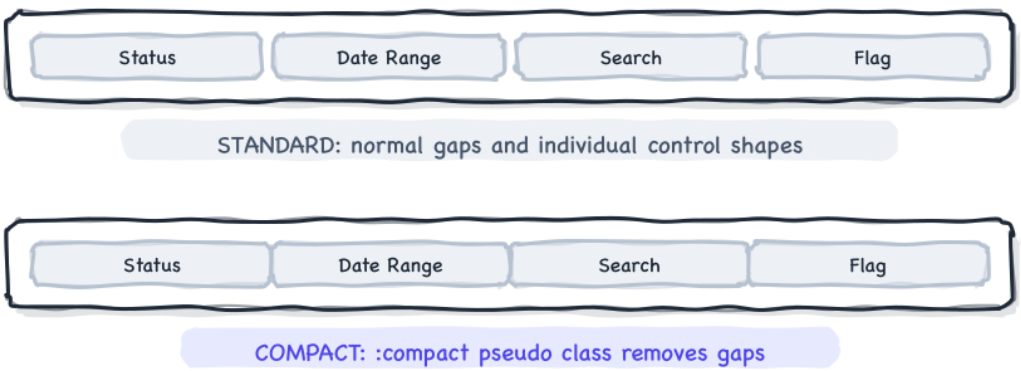
Property	Type	Default	Description
layoutMode	ObjectProperty<LayoutMode>	STANDARD	STANDARD or COMPACT. COMPACT activates the :compact pseudo class.
chips	ReadOnlyListProperty<ChipView<?>>	empty list	Chips representing selected or entered values.
onChange	ObjectProperty<Runnable>	null	Called after supported child controls change; clear() calls it once at the end.

4.1 Factory methods

Method	Result
addSelectionBox(String, Class<Enum>)	SelectionBox with enum values and converter.
addSelectionBox(String, Collection) / varargs	SelectionBox for supplied values.
addCheckBox(String)	CheckBox.
addTextField(String)	CustomTextField.
addSearchTextField(String)	SearchTextField.
addDateRangePicker(String)	DateRangePicker with value null, no preset title and no icon.
addCalendarPicker(String)	Non-editable CalendarPicker with value null.
addDatePicker(String)	Non-editable DatePicker with value null.

5. Layout modes

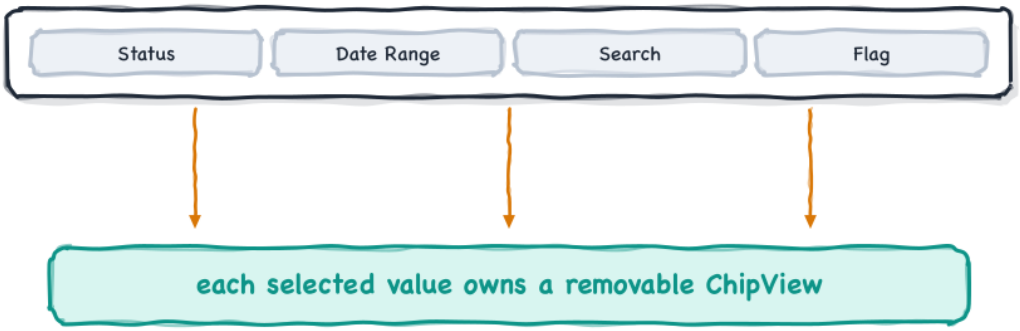
STANDARD keeps normal spacing. COMPACT activates the `:compact` pseudo class and the CSS removes gaps so adjacent filter inputs look like one segmented bar.



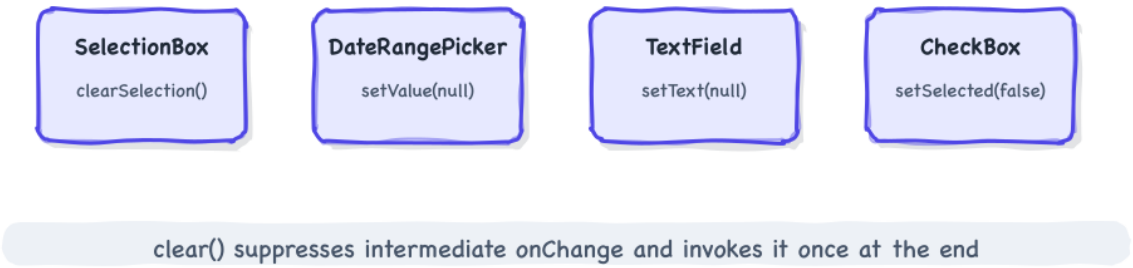
STANDARD and COMPACT layout modes.

6. Chips and clear behaviour

Each supported control contributes chips for its selected or entered values. Closing a chip clears the originating value.



Values flow from controls into the chips list.



clear() resets supported controls and calls onChange once.

Careful. If clear() encounters an unsupported child node type it throws IllegalStateException. Add custom controls only when your application handles clearing them separately.

7. Using values in an application

SimpleFilterView deliberately stops at input collection. The application owns the actual filtering logic and can read the created controls directly inside the onChange callback.

```
SimpleFilterView filters = new SimpleFilterView();
SelectionBox<Status> status = filters.addSelectionBox("Status", Status.class);
CustomTextField search = filters.addSearchTextField("Search");
CheckBox archived = filters.addCheckBox("Archived");

filters.setOnChange(() -> {
    List<Status> statuses = status.getSelectionModel().getSelectedItems();
    String text = search.getText();
    boolean includeArchived = archived.isSelected();
    service.reload(statuses, text, includeArchived);
});
```

Read the factory-created controls when the filter state changes.

- Keep references to the controls returned by the factory methods.
- Debounce expensive back-end calls in your application service if needed.
- Use the chips list for display and clearing; do not parse chip text as application state.
- Call clear() when a saved search or reset button should restore the default state.

8. Styling

Style class / pseudo class	Description
.simple-filter-view	Root HBox.
:compact	Active when layoutMode is COMPACT.
.selection-item	Added to child Regions.
.first, .middle, .last, .only	Position classes for grouped controls.
.check-box	Regular JavaFX CheckBox styling hook.

SimpleFilterView has no styleable CSS properties.

```
.simple-filter-view:compact .selection-item {
    -fx-background-radius: 0;
}
.simple-filter-view .selection-item.only {
    -fx-background-radius: 6px;
}
```

Example CSS for compact grouping.

9. Localization

The control uses ResourceBundleManager only for accessibility. The key `accessible.role-description` has the English default *filter*.

10. Accessibility

SimpleFilterView sets `AccessibleRole.NODE` and uses the localized role description *filter*. The child controls keep their own accessible roles.

11. Recipes

11.1 Use compact mode

```
filters.setLayoutMode(SimpleFilterView.LayoutMode.COMPACT);
```

11.2 Bind chips to a container

```
chips.chipsProperty().bind(filters.chipsProperty());
chips.setOnClear(filters::clear);
```

11.3 Single-selection enum

```
SelectionBox<Status> status = filters.addSelectionBox("Status", Status.class);
status.getSelectionModel().setSelectionMode(SelectionMode.SINGLE);
```

11.4 Reload after change

```
filters.setOnChange(this::reloadResults);
```

11.5 Checklist

1. Use SimpleFilterView for inputs, not automatic data filtering.
2. Use FilterView when predicates and filteredItems are required.
3. Bind chips to a ChipsViewContainer for a clear-all UI.
4. Avoid unsupported custom children unless you handle their clear behaviour.

12. See also

- Demo app: `SimpleFilterViewApp`. Run with `mvn javafx:run -f gemsfx-demo/pom.xml -Dmain.class=com.dlsc.gemsfx.demo.SimpleFilterViewApp`.
- Related controls: FilterView, ChipsViewContainer, SelectionBox, DateRangePicker and CalendarPicker.
- API docs: https://dlsc-software-consulting-gmbh.github.io/GemsFX/api/