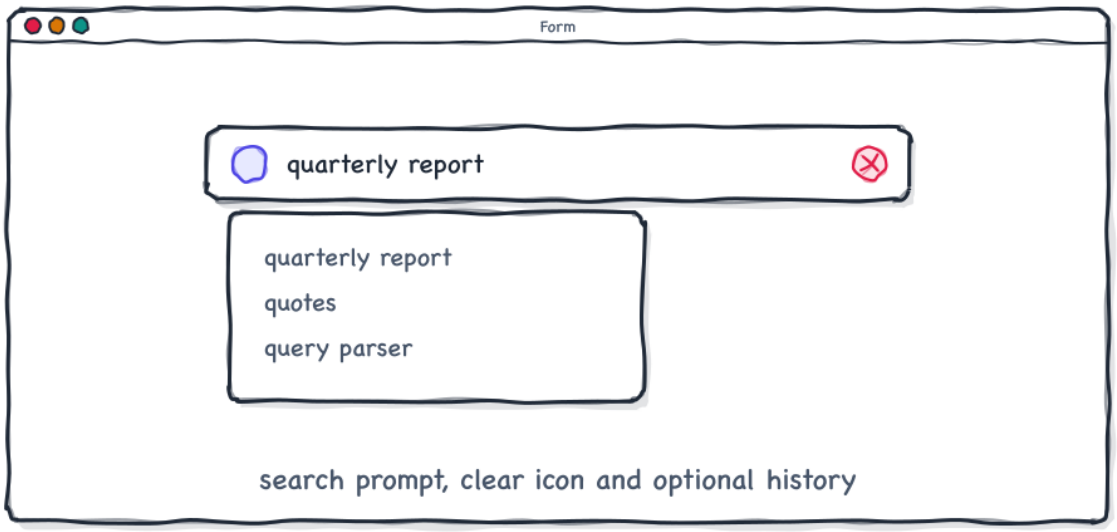
SearchTextField

Developer Manual

GemsFX • com.dlsc.gemsfx • JavaFX Custom Controls



A generated cartoon overview of SearchTextField.

SearchTextField extends CustomTextField with a localized prompt, a search/history button, a clear icon and optional HistoryManager integration.

Table of Contents

1. Introduction

SearchTextField is a search-oriented `CustomTextField`. It shows a search icon on the left, a clear icon on the right when text is present and can connect to a `HistoryManager<String>` for a popup of previous queries.

1.1 Key features

- Localized default prompt and empty-history placeholder.
- Optional persisted history via `StringHistoryManager`.
- Automatic history insertion on Enter and/or focus lost.
- Styleable rounded appearance through `-fx-round`.
- Clear icon visible only when the field has text.

1.2 Maven dependency

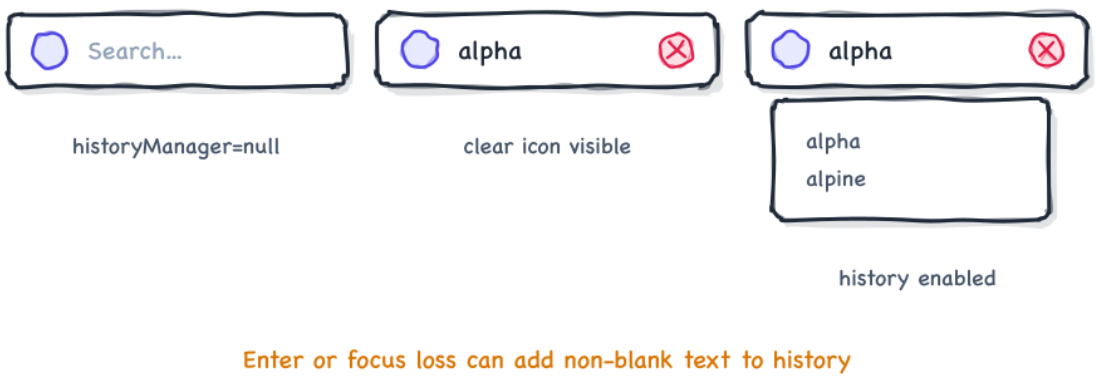
```
<dependency>
    <groupId>com.dlsc.gemsfx</groupId>
    <artifactId>gemsfx</artifactId>
    <version>4.4.1</version>
</dependency>
```

Maven coordinates for the GemsFX control library.

2. Getting started

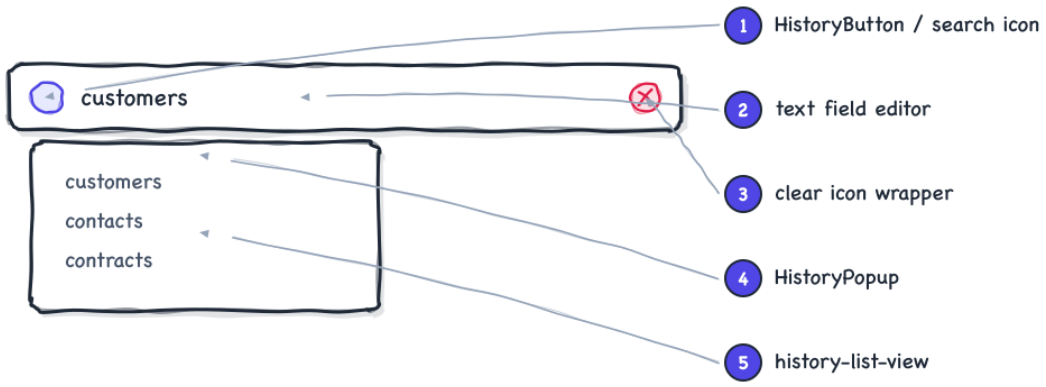
```
SearchTextField field = new SearchTextField();
Preferences prefs = Preferences.userNodeForPackage(MyApp.class);
field.setHistoryManager(new StringHistoryManager(prefs, "main-search"));
field.setRound(true);

field.setOnAction(evt -> runSearch(field.getText()));
```

A complete search field with persisted string history.



Disabled history, clear icon and open history states.

3. Anatomy



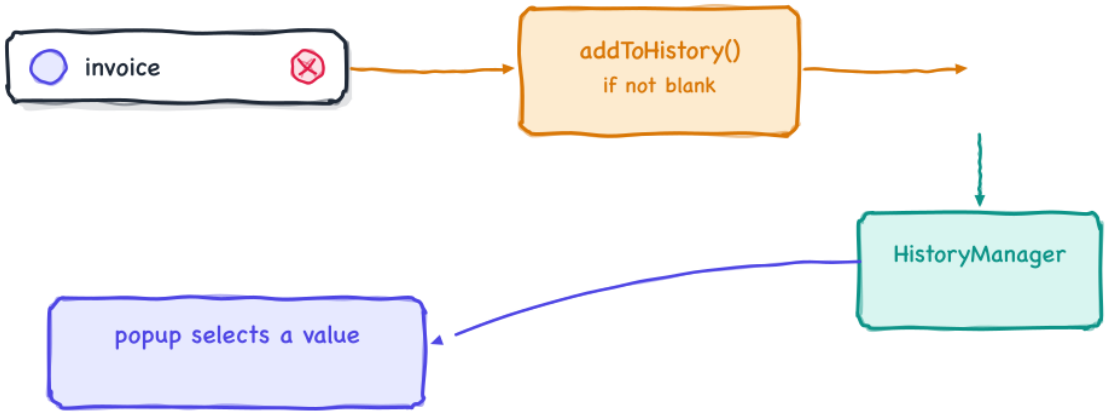
Parts of a SearchTextField.

Part	Node / property	Description
Root	`.search-text-field`	CustomTextField subclass.
History button	`HistoryButton<String>`	Left node; opens the popup when a manager exists.
Clear icon	`.clear-icon-wrapper`	Right node; clears text on click.
Popup	`HistoryPopup`	Owned by HistoryButton and styled by history-button.css.

4. Control API

Property	Type	Default	Description
historyManager	ObjectProperty<HistoryManager<String>>	null	Enables popup history when non-null.
historyPlaceholder	ObjectProperty<Node>	Label("No items.")	Placeholder node for an empty history list.
addingItemToHistoryOnEnter	BooleanProperty	true	Adds non-blank text to history on action events.
addingItemToHistoryOnFocusLost	BooleanProperty	true	Adds non-blank text to history when focus is lost.
round	BooleanProperty	false	Toggles the style class `round`. Styleable.
promptText	StringProperty	Search...	Localized constructor default.

5. History behaviour



How searches are stored and restored.

The field never creates a history manager by itself. When `historyManager` is null, the left button is visually reduced to a search icon and `showPopup()` returns without opening a popup.

When the manager is present, pressing Enter or losing focus calls `addToHistory()` if the corresponding property is true and the text is not blank. Selecting an item in the popup copies it into the field and hides the popup.

```
field.setAddingItemToHistoryOnEnter(false);
field.setAddingItemToHistoryOnFocusLost(true);
field.getHistoryManager().setMaxHistorySize(30);
```

6. Popup customization

History display is delegated to `HistoryButton`. SearchTextField binds the button placeholder and history manager to its own properties, then copies selected values back into the text field.

Extension point	Description
`historyPlaceholder`	Node displayed by the popup list when the manager has no entries.
`HistoryManager`	Controls persistence, maximum size and ordering of history entries.
Inherited `left/right`	Can be replaced, but doing so removes the built-in history or clear affordance.

```
Label empty = new Label("Nothing searched yet");
field.setHistoryPlaceholder(empty);
field.getHistoryManager().setMaxHistorySize(10);
```

Note. The clear icon is a managed node only while text is non-empty, so the field does not reserve right-side space for empty input.

7. Styling

7.1 Style classes and pseudo classes

Selector	Description
`.search-text-field`	root field
`.search-text-field.round`	round root style class
`.history-button`	left button
`.history-button:disabled-popup`	button while history manager is null
`.history-button:popup-showing`	button while popup is visible
`.clear-icon-wrapper`	right clear wrapper
`.history-popup .history-list-view`	popup list from HistoryButton stylesheet

7.2 Styleable CSS properties

CSS property	Type	Default
`-fx-round`	boolean	false

```
.search-text-field {
    -fx-pref-width: 320px;
    -fx-round: true;
}

.search-text-field > .right-pane > .clear-icon-wrapper {
    -fx-padding: 0 8px;
}
```

8. Localization

The constructor loads two strings through `ResourceBundleManager` from `search-text-field.properties`.

Key	English text
`prompt.search`	Search...
`placeholder.history-empty`	No items.

9. Accessibility

The constructor sets `AccessibleRole.TEXT_FIELD`. The embedded `HistoryButton` has button semantics; the clear icon is a clickable graphic, so applications with strict keyboard requirements may replace the right node with a focusable button.

10. Recipes

10.1 Disable automatic history on Enter

```
field.setAddingItemToHistoryOnEnter(false);
```

10.2 Use a custom placeholder

```
field.setHistoryPlaceholder(new Label("No recent searches"));
```

10.3 Clear history

```
Optional.ofNullable(field.getHistoryManager()).ifPresent(HistoryManager::clear);
```

10.4 Checklist

1. Create and set a `HistoryManager` to enable the popup.
2. Use a stable preferences key for persisted history.
3. Bind or style `round` for pill-shaped search boxes.
4. Run the search in the normal `onAction` handler.

11. See also

- Demo application: `com.dlsc.gemsfx.demo.SearchTextFieldApp` (run with `mvn javafx:run -f gemsfx-demo/pom.xml -Dmain.class=com.dlsc.gemsfx.demo.SearchTextFieldApp`)
- `CustomTextField` - superclass for left / right embedded nodes.
- `HistoryButton` - reusable popup button used by the field.
- `EmailField` - text field with validation and suggestions.
- API documentation: https://dlsc-software-consulting-gmbh.github.io/GemsFX/api/