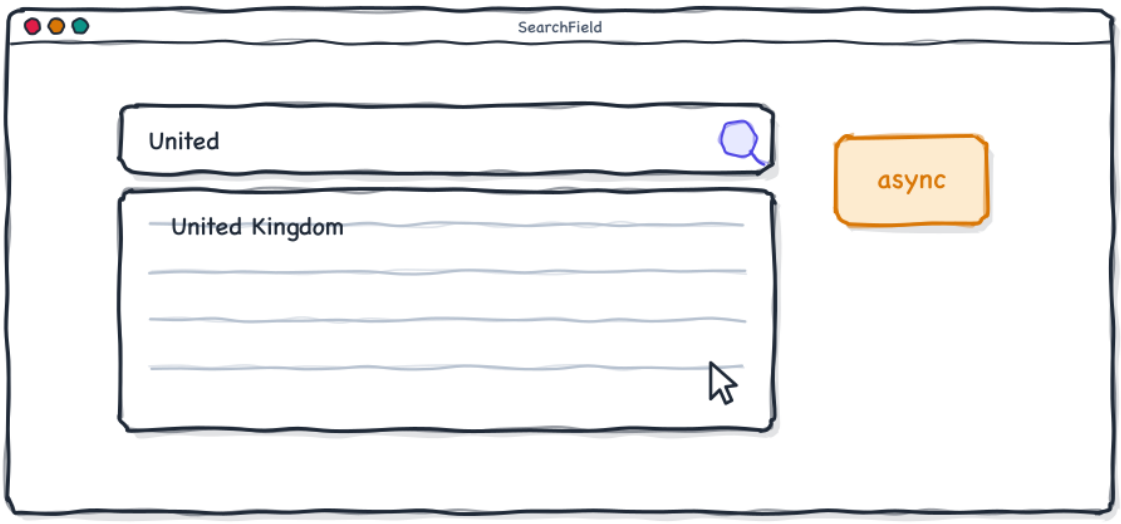
SearchField

Developer Manual

GemsFX • com.dlsc.gemsfx • JavaFX Custom Controls



Generated cartoon overview of SearchField.

A generic control that combines a text editor, asynchronous suggestions, auto-completion, object selection and optional search history. The committed result is a domain object, not just the typed string.

Table of Contents

1. Introduction

SearchField A generic control that combines a text editor, asynchronous suggestions, auto-completion, object selection and optional search history. The committed result is a domain object, not just the typed string.

1.1 Key features

- Text changes restart a JavaFX Service with a 250 ms delay.
- ENTER or RIGHT commits the selected item.
- API and styling details below are verified against the control, skin, CSS and resource bundles.

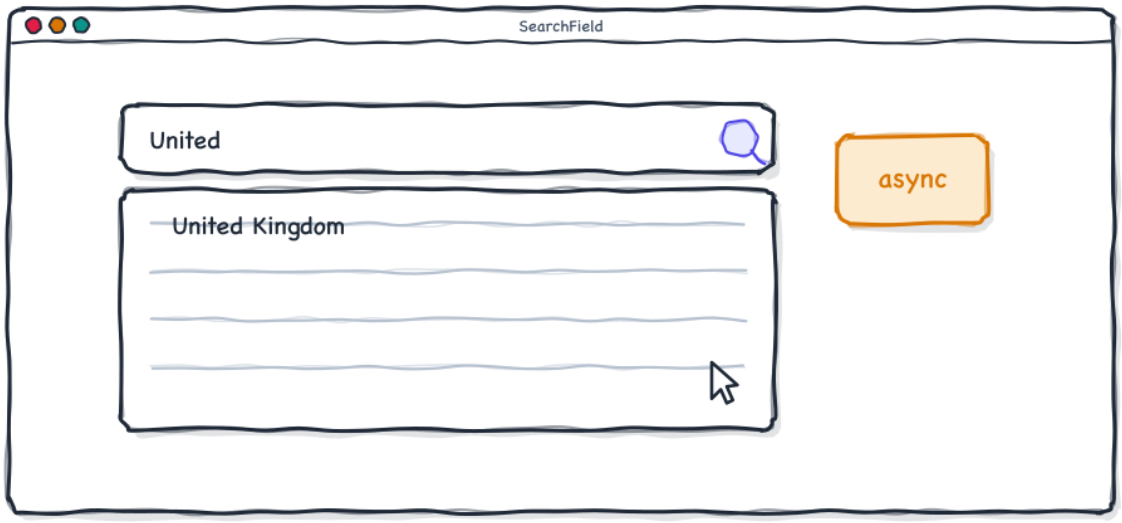
1.2 Maven dependency

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

2. Getting started

```
SearchField<Country> field = new SearchField<>();
field.setSuggestionProvider(request -> countries.stream().filter(c -> c.name().contains(requ
est.getUserText())).toList());
field.setConverter(countryConverter);
field.setMatcher((country, text) -> country.name().startsWith(text));
field.setComparator(Comparator.comparing(Country::name));
field.setOnCommit(System.out::println);
```

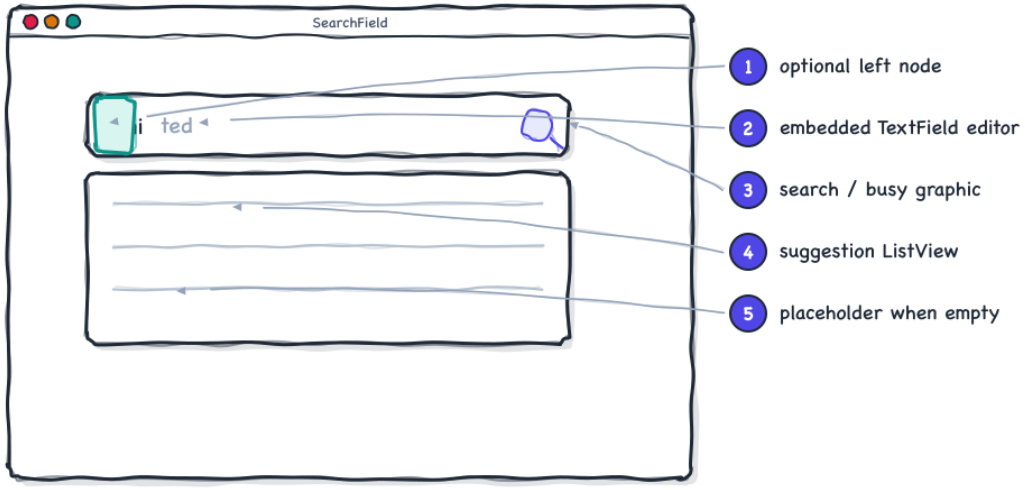
A compact setup for SearchField.



SearchField in a typical application context.

3. Anatomy

The diagram identifies the nodes and state holders created by the implementation.



The parts of SearchField.

Topic	Verified source detail
Stylesheet	`com/dlsc/gemsfx/search-field.css`
Root style class	`.search-field`
Graphics	Generated SVGs; no screenshots.

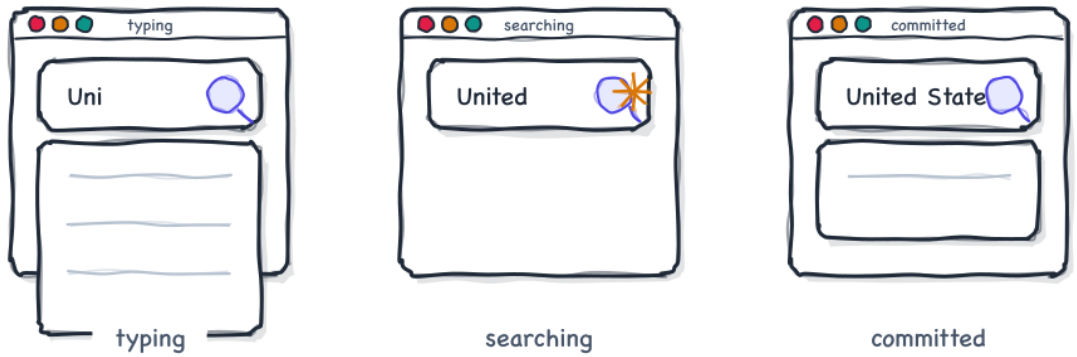
4. Control API

4.1 Properties and callbacks

Property	Type	Default	Description
suggestionProvider	ObjectProperty<Callback<SearchFieldSuggestionRequest, Collection<T>>>	null	Callback used by the background task to find suggestions.
suggestions	ObservableList<T>	empty, read-only	Current unmodifiable suggestion list.
selectedItem	ObjectProperty<T>	null	Currently selected item.
newItemProducer	ObjectProperty<Callback<String,T>>	null	Creates a value when no suggestion matches.
matcher	ObjectProperty<BiFunction<T,String,Boolean>>	converter text starts with search text	Finds the best match.
converter	ObjectProperty<StringConverter<T>>	item.toString() or empty	Turns items into display text.

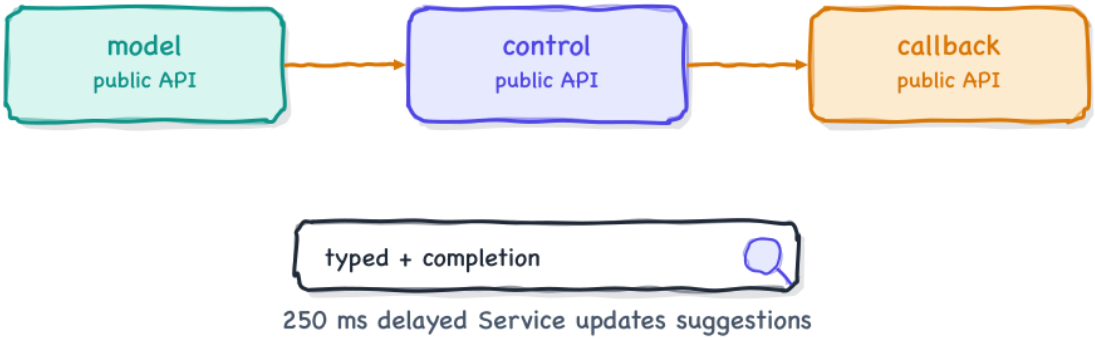
Property	Type	Default	Description
text / promptText	StringProperty	empty string	Editor text and prompt.
graphic / busyGraphic	ObjectProperty<Node>	MDI_MAGNIFY / MDI_CACHED	Idle and searching graphics.
historyManager	ObjectProperty<HistoryManager<String>>	null	Enables history.
autoCommitOnFocusLost	BooleanProperty	true	Commits on focus lost.
addingItemToHistoryOnEnter / OnFocusLost / OnCommit	BooleanProperty	true	Controls history writes.
onCommit	ObjectProperty<Consumer<T>>	null	Invoked on user commit.

5. Behaviour



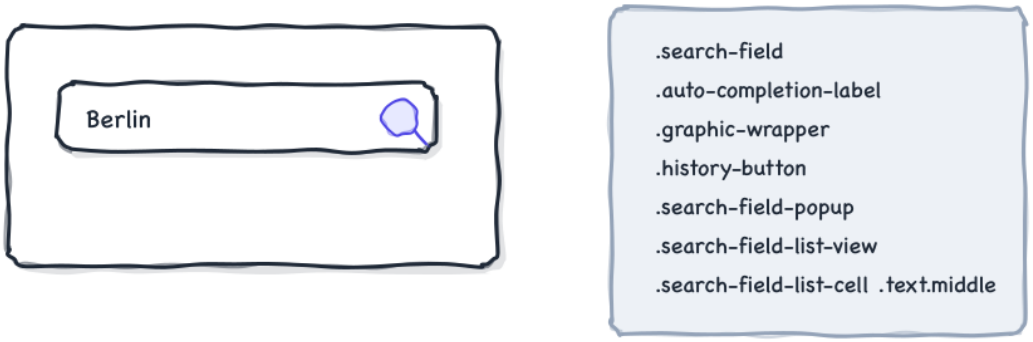
Important runtime states of SearchField.

- Text changes restart a JavaFX Service with a 250 ms delay.
- ENTER or RIGHT commits the selected item.
- ESCAPE cancels, clears the field and hides popups.
- History is disabled while historyManager is null.



How application data flows through SearchField.

6. Styling



Style hooks for SearchField.

6.1 Style classes and pseudo classes

Selector / pseudo class	Purpose
`.search-field`	Verified in source, skin or CSS.
`.auto-completion-label`	Verified in source, skin or CSS.
`.graphic-wrapper`	Verified in source, skin or CSS.
`.history-button`	Verified in source, skin or CSS.
`.search-field-popup`	Verified in source, skin or CSS.
`.search-field-list-view`	Verified in source, skin or CSS.
`.search-field-list-cell .text.middle`	Verified in source, skin or CSS.
`:disabled-popup`	Verified in source, skin or CSS.

Selector / pseudo class	Purpose
`:left-node-visible / :right-node-visible / :no-side-nodes`	Verified in source, skin or CSS.

6.2 Styleable CSS properties

CSS property	Type	Default
`-fx-hide-popup-with-single-choice`	boolean	false
`-fx-hide-popup-with-no-choice`	boolean	false
`-fx-auto-completion-gap`	size	1
`-fx-show-search-icon`	boolean	true
`-fx-auto-commit-on-focus-lost`	boolean	true
`-fx-adding-item-to-history-on-enter`	boolean	true
`-fx-adding-item-to-history-on-focus-lost`	boolean	true
`-fx-adding-item-to-history-on-commit`	boolean	true

```
.search-field {
    /* start with the documented root selector */
}
```

7. Localization

Key	English text
`placeholder.history-empty`	No items.
`placeholder.suggestions-empty`	No items found

8. Accessibility

Sets AccessibleRole.TEXT_FIELD. No automatic accessible text binding is installed.

9. Recipes

9.1 Programmatic configuration

```
SearchField<Country> field = new SearchField<>();
field.setSuggestionProvider(request -> countries.stream().filter(c -> c.name().contains(requ
est.getUserText())).toList());
field.setConverter(countryConverter);
field.setMatcher((country, text) -> country.name().startsWith(text));
field.setComparator(Comparator.comparing(Country::name));
field.setOnCommit(System.out::println);
```

9.2 Checklist

1. Use the public properties listed in the API chapter.
2. Style through documented selectors and CSS properties only.
3. Keep application model updates in callbacks such as onClose, onClear or onItemSelected.
4. Do not depend on private skin nodes except for documented style selectors.

10. See also

- Demo application: `com.dlsc.gemsfx.demo.SearchFieldApp` (run with `mvn javafx:run -f gemsfx-demo/pom.xml -Dmain.class=com.dlsc.gemsfx.demo.SearchFieldApp`)
- API documentation: https://dlsc-software-consulting-gmbh.github.io/GemsFX/api/