

Manuale di KatePart

**Thad McGinnis
Anne-Marie Mahfouf
Anders Lund
T.C. Hollingsworth
Christoph Cullmann
Lauri Watts**

**Traduzione della documentazione: Marco Poletti
Traduzione della documentazione: Samuele Kaplun
Traduzione della documentazione:**



Indice

1	Introduzione	8
2	Alcune cose fondamentali	9
2.1	Drag and Drop (Trascinamento)	9
2.2	Scorciatoie	9
3	Lavorare con l'editor KatePart	12
3.1	Panoramica	12
3.2	Navigare nel testo	13
3.3	Lavorare con la selezione	13
3.3.1	Usare la selezione a blocchi	14
3.3.2	Usare l'opzione Sovrascrivi selezione	14
3.3.3	Usare la selezione permanente	14
3.4	Copiare ed incollare il testo	15
3.5	Trovare e sostituire il testo	15
3.5.1	Le barre di ricerca e di sostituzione	15
3.5.2	Trovare del testo	16
3.5.3	Sostituire il testo	16
3.6	Usare i segnalibri	17
3.7	Andare a capo automaticamente	17
3.8	Usare il rientro automatico	18
3.9	Indicatori di modifica delle righe	18
3.10	La mini-mappa di scorrimento	19
3.11	Cursori multipli	20
3.11.1	Creazione di più cursori	20
3.11.2	Lavorare con più cursori	20
4	Le voci dei menu	21
4.1	Il menu File	21
4.2	Il menu Modifica	22
4.3	Il menu Selezione	24
4.4	Il menu Visualizza	26
4.5	Il menu Vai	27
4.6	Il menu Strumenti	28
4.7	I menu Impostazioni e Aiuto	31

5	Strumenti avanzati di modifica	32
5.1	Commenta/Decommenta	32
5.2	La riga di comando per il componente dell'editor	32
5.2.1	I comandi standard della riga di comando	33
5.2.1.1	Comandi per la configurazione dell'editor	33
5.2.1.2	Comandi di modifica	35
5.2.1.3	Comandi per gli spostamenti	39
5.2.1.4	Comandi per le funzioni fondamentali dell'editor (dipendono dall'applicazione in cui il componente editor è in uso)	40
5.3	Uso del raggruppamento del codice	41
6	Estensione di KatePart	42
6.1	Introduzione	42
6.2	Lavorare con l'evidenziazione della sintassi	42
6.2.1	Panoramica	42
6.2.2	Il sistema di evidenziazione della sintassi di KatePart	43
6.2.2.1	Come funziona	43
6.2.2.2	Regole	44
6.2.2.3	Stili di contesto e parole chiave	44
6.2.2.4	Stili predefiniti	44
6.2.3	Il formato XML di definizione dell'evidenziazione	45
6.2.3.1	Panoramica	45
6.2.3.2	Le sezioni in dettaglio	48
6.2.3.3	Stili predefiniti disponibili	50
6.2.4	Regole di rilevamento dell'evidenziazione	51
6.2.4.1	Le regole in dettaglio	53
6.2.4.2	Suggerimenti e trucchi	57
6.3	Lavorare con i temi di colori	58
6.3.1	Panoramica	58
6.3.2	I temi di colori di KSyntaxHighlighting	59
6.3.3	Il formato JSON dei temi di colori	60
6.3.3.1	Panoramica	60
6.3.3.2	La struttura JSON	61
6.3.3.3	Struttura principale dei file JSON di un tema di colori	61
6.3.3.4	Metadati	63
6.3.4	I colori in dettaglio	64
6.3.4.1	Modificare i colori	64
6.3.4.2	Stili di testo predefiniti	70
6.3.4.3	Stili di evidenziazione del testo personalizzati	73
6.3.5	L'GUI temi di colori	74
6.3.5.1	Creare un nuovo tema	75
6.3.5.2	Importare o esportare i file di tema JSON	75

6.3.5.3	Modificare i temi di colori	75
6.3.5.3.1	Colori	75
6.3.5.3.2	Stili di testo predefiniti	75
6.3.5.3.3	Stili di testo evidenziato	76
6.3.6	Suggerimenti e trucchi	76
6.3.6.1	Contrasto dei colori del testo	76
6.3.6.2	Suggerimenti di consistenza per l'evidenziazione della sintassi	76
6.4	Scripting con JavaScript	77
6.4.1	Script di rientro	77
6.4.1.1	L'intestazione dello script di rientro	77
6.4.1.2	Il codice sorgente del rientratore	78
6.4.2	Script da riga di comando	79
6.4.2.1	L'intestazione dello script per la riga di comando	79
6.4.2.2	Il codice sorgente dello script	80
6.4.2.2.1	Stabilire le scorciatoie	81
6.4.3	API per gli script	82
6.4.3.1	Cursori e intervalli	82
6.4.3.1.1	Il prototipo dei cursori	82
6.4.3.1.2	Il prototipo degli intervalli	83
6.4.3.2	Funzioni globali	84
6.4.3.2.1	Leggere e includere file	84
6.4.3.2.2	Debug	85
6.4.3.2.3	Traduzione	85
6.4.3.3	API della vista	85
6.4.3.4	API del documento	87
6.4.3.5	L'API dell'editor	92
7	Configura KatePart	93
7.1	Configurazione del componente editor	93
7.1.1	Aspetto	93
7.1.1.1	Generale	93
7.1.1.2	Bordi	95
7.1.2	Temi di colori	96
7.1.3	Modifica	96
7.1.3.1	Generale	96
7.1.3.2	Navigazione del testo	97
7.1.3.3	Rientro	98
7.1.3.4	Completamento delle parole	99
7.1.3.5	Controllo ortografico	100
7.1.3.6	Modalità di inserimento Vi	100
7.1.4	Apri e salva	100
7.1.4.1	Generale	100

Manuale di KatePart

7.1.4.2	Avanzate	101
7.1.4.3	Modi e tipi di file	102
7.2	Configurazione con le variabili dei documenti	103
7.2.1	Come KatePart usa le variabili	104
7.2.2	Variabili disponibili	104
7.2.3	Opzioni estese nei file <code>.kateconfig</code>	107
8	Riconoscimenti e licenza	108
9	La modalità di inserimento Vi	110
9.1	Modalità di inserimento Vi	110
9.1.1	Incompatibilità con Vim	110
9.1.2	Cambiare modalità	111
9.1.3	Integrazione con le funzionalità di Kate	111
9.1.4	Comandi supportati nelle modalità normale e visuale	112
9.1.5	Movimenti supportati	113
9.1.6	Oggetti di testo supportati	115
9.1.7	Comandi supportati nella modalità di inserimento	115
9.1.8	L'oggetto di testo tra virgole	116
9.1.9	Funzionalità mancanti	116
A	Le espressioni regolari	117
A.1	Introduzione	117
A.2	I modelli	118
A.2.1	I caratteri speciali	118
A.2.2	Classi di caratteri ed abbreviazioni	118
A.2.2.1	Caratteri con un significato speciale all'interno delle classi di caratteri	120
A.2.3	Alternative: corrispondenza del tipo 'uno fra'	120
A.2.4	I sotto-modelli	120
A.2.4.1	Specificare delle alternative	120
A.2.4.2	Cattura del testo corrispondente (riferimenti all'indietro)	120
A.2.4.3	Asserzioni di lookahead	121
A.2.4.4	Asserzioni di Lookbehind	121
A.2.5	Caratteri con un significato speciali in un modello	121
A.3	Quantificatori	122
A.3.1	Ingordigia	122
A.3.2	Esempi in contesto	123
A.4	Le asserzioni	123
B	Indice analitico	125

Sommario

KatePart è un componente editor con ogni funzionalità dalla comunità KDE.

Capitolo 1

Introduzione

KatePart è un editor di testo con ogni funzionalità usato da molte applicazioni di Qt™ e KDE. KatePart è più di un editor di testo; è pensato per essere un editor per programmatori e potrebbe essere considerato come una parziale alternativa a editor più potenti. Una delle principali funzionalità di KatePart è la colorazione della sintassi, adattata a diversi linguaggi di programmazione quali: C/C++, Java™, Python, Perl, Bash, Modula 2, HTML e Ada.

KWrite è un semplice editor di testo basato su KatePart. Ha un'interfaccia a documento singolo (*Single Document Interface*, o 1SDI), che permette di modificare un file alla volta in ogni finestra. Siccome KWrite è un'implementazione molto semplice di KatePart, non richiede una sua propria documentazione. Se sai usare KWrite, puoi usare KatePart ovunque!

Capitolo 2

Alcune cose fondamentali

KWrite e molti altri programmi che usano KatePart sono molto semplici da usare. Chiunque abbia già usato un editor di testo non dovrebbe incontrare problemi.

2.1 Drag and Drop (Trascinamento)

KatePart usa il protocollo Drag and Drop di KDE. I file possono essere trascinati e rilasciati su KatePart dal Desktop, Dolphin o da qualche sito FTP remoto aperto in una delle finestre di Dolphin.

2.2 Scorciatoie

Molte delle scorciatoie sono configurabili attraverso il menu [Impostazioni](#). Per impostazione predefinita KatePart risponde alle seguenti scorciatoie:

Ins	Alterna tra la modalità Inserisci e Sovrascrivi. Quando si è in modalità di inserimento l'editor aggiungerà qualsiasi carattere digitato spostando a destra i caratteri dopo il cursore. Con la modalità di sovrascrittura ogni carattere inserito elimina quello immediatamente alla destra del cursore.
←	Sposta il cursore a sinistra di un carattere.
→	Sposta il cursore a destra di un carattere.
↑	Sposta il cursore in su di una riga.
↓	Sposta il cursore in giù di una riga.
Ctrl+E	Vai alla posizione di modifica precedente nel documento.
Ctrl+Shift+E	Vai alla posizione di modifica successiva nel documento.
Alt+Shift+↑	Sposta il cursore al rientro corrispondente precedente.
Alt+Shift+↓	Sposta il cursore al rientro corrispondente precedente.

Ctrl+6	Vai alla parentesi corrispondente.
Pag↑	Sposta il cursore in su di una pagina.
Pag↓	Sposta il cursore in giù di una pagina.
&#8598;	Sposta il cursore all'inizio della riga.
Fine	Sposta il cursore alla fine della riga.
Ctrl+&#8598;	Vai all'inizio del documento.
Ctrl+Fine	Vai alla fine del documento.
Ctrl+↑	Scorri una riga in su.
Ctrl+↓	Scorri una riga in giù.
Ctrl+→	Sposta parola a destra.
Ctrl+←	Sposta parola a sinistra.
Ctrl+Shift+↑	Sposta righe in alto.
Ctrl+Shift+↓	Sposta righe in basso.
Ctrl+.	Duplica le righe selezionate in basso.
Ctrl+B	Imposta un segnalibro.
Alt+Pag↑	Segnalibro precedente.
Alt+Pag↓	Segnalibro successivo.
Canc	Cancella il carattere alla destra del cursore (o il testo selezionato).
Backspace	Cancella il carattere a sinistra del cursore.
Ctrl+Canc	Cancella parola a destra.
Ctrl+Backspace	Cancella parola a sinistra.
Ctrl+K	Cancella riga.
Shift+Invio	Inserisce una nuova riga aggiungendovi i caratteri iniziali della riga corrente che non sono lettere o numeri. È utile, ad esempio, per scrivere commenti nel codice. Alla fine della riga '//' del testo' premi la scorciatoia e la prossima riga inizierà già con '// '. In questo modo con si devono sempre inserire i caratteri per il commento all'inizio di ogni nuova riga con commenti.
Ctrl+Shift+Invio	Crea una nuova riga sotto l'attuale.
Ctrl+Alt+Invio	Crea una nuova riga sopra l'attuale.
Shift+←	Seleziona un carattere a sinistra nel testo.
Shift+→	Seleziona un carattere a destra nel testo.
Ctrl+F	Trova.
F3	Trova successivo .
Shift+F3	Trova precedente.
Ctrl+H	Trova selezionato.
Ctrl+Shift+H	Trova selezionato all'indietro.
Ctrl+Shift+→	Seleziona parola a destra.
Ctrl+Shift+←	Seleziona parola a sinistra.
Shift+&#8598;	Seleziona fino all'inizio della riga.
Shift+Fine	Seleziona fino alla fine della riga.
Shift+↑	Seleziona fino alla riga precedente.
Shift+↓	Seleziona fino alla riga successiva.
Ctrl+Shift+6	Seleziona fino alla parentesi corrispondente.
Ctrl+Shift+Pag↑	Seleziona fino all'inizio della vista.
Ctrl+Shift+Pag↓	Seleziona fino alla fine della vista.
Shift+Pag↑	Seleziona fino alla pagina precedente.
Shift+Pag↓	Seleziona fino alla pagina successiva.
Ctrl+Shift+&#8598;	Seleziona fino all'inizio del documento.

Ctrl+Shift+Fine	Seleziona fino alla fine del documento.
Ctrl+&#8598;	Seleziona tutto.
Ctrl+Shift+A	Deseleziona.
Ctrl+Shift+B	Modalità di selezione a blocchi.
Ctrl+C / Ctrl+Ins	Copia il testo selezionato negli appunti.
Ctrl+D	Commenta.
Ctrl+Shift+D	Decommenta.
Ctrl+G	Va alla riga...
Ctrl+I	Fai rientrare la selezione.
Ctrl+Shift+I	Annulla rientro alla selezione.
Ctrl+J	Unisci righe.
Ctrl+P	Stampa.
Ctrl+R	Sostituisci.
Ctrl+S	Invoca il comando Salva.
Ctrl+Shift+S	Salva come.
Ctrl+U	Maiuscolo.
Ctrl+Shift+U	Minuscolo.
Ctrl+Alt+U	Iniziali maiuscole.
Ctrl+V / Shift+Ins	Incolla il testo degli appunti nella riga di edit.
Ctrl+X / Shift+Ins	Cancella il testo selezionato e lo copia negli appunti.
Ctrl+Z	Annulla.
Ctrl+Shift+Z	Rifai.
Ctrl+-	Diminuisce carattere.
Ctrl++Ctrl+=	Ingrandisce carattere.
Ctrl+Shift+-	Contrai nodi di massimo livello.
Ctrl+Shift++	Espandi nodi di massimo livello.
Ctrl+Spazio	Invoca completamento del codice.
F5	Ricarica.
F6	Mostra/Nascondi bordo per le icone.
F7	Passa alla riga di comando.
F9	Mostra segni di raggruppamento.
F10	A capo automatico dinamico.
F11	Mostra/Nascondi i numeri di riga.
Ctrl+T	Trasponi caratteri.
Ctrl+Shift+O	Controllo ortografico automatico.
Ctrl+Shift+V	Passa alla modalità di inserimento successiva.
Ctrl+8	Riusa parola sopra.
Ctrl+9	Riusa parola sotto.
Ctrl+Alt+#	Espandi abbreviazione.
Ctrl+Alt+↑	Aggiungi un cursore sopra l'attuale.
Ctrl+Alt+↓	Aggiungi un cursore sotto l'attuale.
Shift+Alt+I	Crea un cursore alla fine di ogni riga nella selezione.
Alt+J	Trova la prossima occorrenza della parola sotto il cursore e selezionala.
Ctrl+Alt+Shift+J	Cerca tutte le occorrenze della parola sotto il cursore e selezionala.

Capitolo 3

Lavorare con l'editor KatePart

Anders Lund
Dominik Haumann
Aggiornamento: Zamponi
Traduzione: Luigi Toscano
Manutenzione della documentazione: Federico Zenith

3.1 Panoramica

L'editor KatePart è l'area modificabile della finestra di KatePart. Questo editor è condiviso tra Kate e KWrite, e può essere usato anche da Konqueror per mostrare i file di testo presenti nel tuo computer o in rete.

L'editor è costituito dai seguenti componenti:

L'area di modifica

Qui si trova il testo del documento.

Le barre di scorrimento

Le barre di scorrimento indicano la posizione della parte visibile del testo del documento, e possono essere utilizzate per spostarsi nel documento. L'operazione di trascinamento delle barre di scorrimento non modifica la posizione del cursore di inserimento.

Le barre di scorrimento sono mostrate e nascoste secondo le necessità.

Il bordo per le icone

Il bordo per le icone è un piccolo pannello alla sinistra dell'editor che mostra una piccola icona in corrispondenza delle righe marcate.

Puoi impostare o rimuovere un [segnalibro](#) in una riga visibile facendo clic con il tasto sinistro del mouse nel bordo per le icone in corrispondenza di tale riga.

La visualizzazione del bordo per le icone può essere attivata o disattivata tramite la voce di menu **Visualizza** → **Mostra bordo per le icone**.

Il pannello per i numeri di riga

Il pannello per i numeri di riga mostra i numeri di riga di tutte le righe visibili nel documento.

La visualizzazione del pannello per i numeri di riga può essere attivata o disattivata tramite la voce di menu **Visualizza** → **Mostra i numeri di riga**.

Il pannello di raggruppamento

Il pannello di raggruppamento ti consente di richiudere o di espandere i blocchi raggruppati di righe. Le regioni richiudibili vengono determinate in base alle regole specificate nelle definizioni di evidenziazione della sintassi per il documento.

ALTRI ARGOMENTI IN QUESTO CAPITOLO:

- [Navigare nel testo](#)
- [Lavorare con la selezione](#)
- [Copiare ed incollare il testo](#)
- [Trovare e sostituire il testo](#)
- [Usare i segnalibri](#)
- [Andare a capo automaticamente](#)
- [Usare il rientro automatico](#)

3.2 Navigare nel testo

Lo spostamento nel testo in KatePart avviene come nella maggior parte degli editor di testo grafici. Puoi spostare il cursore tramite i tasti freccia e i tasti **Pag**↑, **Pag**↓, **↖** e **Fine**, in combinazione con i modificatori **Ctrl** e **Shift**. Il tasto **Shift** viene sempre utilizzato per generare una selezione, mentre **Ctrl** provoca effetti diversi a seconda del tasto:

- Per i tasti ↑ e ↓ significa lo scorrimento della pagina invece dello spostamento del cursore.
- Per i tasti ← e → significa saltare le parole invece dei caratteri.
- Per i tasti **Pag**↑ e **Pag**↓ significa spostare il cursore ai limiti della vista invece di navigare nel testo.
- Per i tasti **↖** e **Fine** significa spostare il cursore all’inizio o alla fine del documento invece che all’inizio o alla fine della riga.

KatePart ti mette anche a disposizione un modo per saltare rapidamente ad una parentesi corrispondente: metti il cursore all’interno di una parentesi e premi **Ctrl-6** per saltare alla parentesi corrispondente.

Inoltre puoi usare i [segnalibri](#) per saltare rapidamente a delle posizioni che puoi definire a tua scelta.

3.3 Lavorare con la selezione

Ci sono due metodi di base per selezionare del testo in KatePart: tramite l’uso rispettivamente del mouse e della tastiera.

Per selezionare usando il mouse tieni premuto il tasto sinistro del mouse mentre trascini il cursore del mouse dal punto iniziale della selezione fino al punto finale. Il testo sarà selezionato mentre trascini.

Facendo doppio clic su una parola questa verrà selezionata.

Facendo clic triplo su una riga questa verrà selezionata interamente.

Se il tasto **Shift** viene tenuto premuto mentre si fa clic, il testo sarà selezionato:

- Se non è stato ancora selezionato alcun testo, dalla posizione del cursore del testo alla posizione del cursore del mouse.
- Se è già attiva una selezione, a partire e includendo tale selezione fino alla posizione del cursore del mouse.

NOTA

Quando selezioni il testo tramite il trascinamento del mouse, il testo selezionato viene copiato negli appunti, e può essere incollato tramite un clic del pulsante centrale del mouse nell'editor o in qualunque altra applicazione in cui vuoi incollarlo.

Per selezionare con la tastiera tieni premuto il tasto **Shift** mentre utilizzi i tasti di navigazione (i tasti freccia, **Page Up**, **Page Down**, **Home** e **End**, anche in combinazione con **Ctrl** per estendere il movimento del cursore di testo).

Vedi anche la sezione [Navigare nel testo](#) in questo capitolo.

Per Copiare la selezione corrente usa la voce di menu **Modifica** → **Copia**, o la scorciatoia di tastiera (quella predefinita è **Ctrl+C**).

Per deselezionare la selezione corrente usa la voce di menu **Modifica** → **Deseleziona**, o la scorciatoia di tastiera (quella predefinita è **Ctrl+Shift+A**), o fai clic con il tasto sinistro del mouse nell'editor.

3.3.1 Usare la selezione a blocchi

Quando la selezione a blocchi è attiva puoi effettuare ‘selezioni verticali’ nel testo, cioè selezionare colonne ben definite da righe differenti. Questo è comodo ad esempio per lavorare con righe con dati separati da tabulazioni.

La selezione a blocchi può essere attivata o disattivata tramite la voce di menu **Modifica** → **Modalità selezione a blocchi**. La scorciatoia di tastiera predefinita è **Ctrl+Shift+B**.

3.3.2 Usare l'opzione Sovrascrivi selezione

Se l'opzione Sovrascrivi selezione è attiva il testo presente nella selezione verrà sostituito se si scrive o si incolla del testo nella selezione. Se non è attiva il nuovo testo verrà aggiunto alla posizione del cursore di testo.

L'opzione Sovrascrivi selezione è attiva come impostazione predefinita.

Per cambiare l'impostazione di questa opzione usa la pagina **cursore e selezione** della **finestra di configurazione** .

3.3.3 Usare la selezione permanente

Quando la selezione permanente è attiva la selezione corrente non verrà annullata se si digitano dei caratteri o si muove il cursore. Questo vuol dire che puoi spostare il cursore dalla selezione e continuare a digitare.

La selezione permanente è disattivata come impostazione predefinita.

La selezione permanente può essere abilitata tramite la pagina **cursore e selezione** della **finestra di configurazione**.

ATTENZIONE

Se la selezione permanente e Sovrascrivi selezione sono entrambe attive, digitare o incollare del testo quando il cursore di testo è all'interno della selezione farà sì che questa venga annullata, e il testo contenuto sostituito.

3.4 Copiare ed incollare il testo

Per copiare del testo selezionato, ed usa la voce di menu **Modifica** → **Copia**. Inoltre, se selezioni del testo con il mouse, questo verrà copiato nella selezione di X.

Per incollare il testo presente negli appunti usa la voce di menu **Modifica** → **Incolla**.

Inoltre il testo selezionato tramite il mouse può essere incollato facendo clic con il tasto centrale del mouse del mouse nella posizione desiderata.

SUGGERIMENTO

Se utilizzi l'ambiente desktop KDE puoi recuperare il testo copiato precedentemente da una qualsiasi applicazione tramite l'icona di Klipper nel vassoio di sistema.

3.5 Trovare e sostituire il testo

3.5.1 Le barre di ricerca e di sostituzione

KatePart ha una barra di ricerca incrementale ed una barra di ricerca e di sostituzione potenziata, che permette di inserire una stringa di sostituzione oltre ad alcune altre opzioni.

Le barre presentano le seguenti opzioni comuni:

Trova

Qui va inserita la stringa di ricerca. L'interpretazione della stringa dipende da alcune delle opzioni descritte più avanti.

Distingui le maiuscole

Se abilitato la ricerca sarà limitata alle corrispondenze con le stesse maiuscole nello schema di ricerca.

La barra potenziata di ricerca e di sostituzione mette a disposizione alcune altre opzioni:

Testo semplice

Cerca le corrispondenze letterali della stringa di ricerca.

Parole intere

Se selezionato la ricerca avrà successo solo se c'è un delimitatore di parola ad entrambi gli estremi della stringa trovata, cioè un carattere non alfanumerico: qualche altro carattere visibile o un fine riga.

Sequenze di escape

Se selezionato l'elemento **Aggiungi** sarà abilitato in fondo al menu contestuale delle caselle di testo, così potrai aggiungere le sequenze di escape nello schema di ricerca da un elenco predefinito.

Espressione regolare

Se selezionato la stringa di ricerca viene interpretata come un'espressione regolare. L'elemento **Aggiungi** sarà abilitato in fondo al menu contestuale delle caselle di testo, così potrai aggiungere un'espressione regolare allo schema di ricerca da un elenco predefinito.

Vedi la sezione [Espressioni regolari](#) per maggiori informazioni.

Cerca solo all'interno della selezione

Se abilitato la ricerca e la sostituzione saranno effettuate solo nel testo selezionato.

Trova tutti

Fare clic su questo pulsante evidenzia tutte le corrispondenze nel documento, e ne mostra il numero in una finestrella a comparsa.

3.5.2 Trovare del testo


Per trovare del testo avvia la barra di ricerca incrementale con **Ctrl+F** o dall'elemento del menu **Modifica** → **Trova...**


Ciò apre la barra di ricerca incrementale nella parte bassa della finestra dell'editor. Nella parte sinistra della barra c'è un pulsante con un'icona per chiudere la barra, seguito da una piccola casella di testo per inserire la chiave di ricerca.

Quando inizi ad inserire la chiave di ricerca, la ricerca inizia immediatamente. Se c'è una corrispondenza nel testo questa viene evidenziata, e lo sfondo della casella di testo diventa verde chiaro. Se la ricerca non dà risultati, lo sfondo della casella di testo diventa rosso chiaro.

Usa il pulsante  oppure  per andare alla corrispondenza precedente o successiva nel documento.

Le corrispondenze nel documento sono evidenziate anche quando chiudi la barra di ricerca; premi il tasto **Esc** per togliere questa evidenziazione.

Puoi scegliere se la ricerca debba distinguere tra maiuscole e minuscole: selezionando  limiterai la ricerca alle corrispondenze con le stesse maiuscole e minuscole dello schema di ricerca.

Fai clic sul pulsante  nella parte destra della barra di ricerca incrementale per usare la barra potenziata di ricerca e sostituzione.

Per ripetere l'ultima operazione di ricerca, se ce n'è una, senza richiamare la barra di ricerca incrementale, usa **Modifica** → **Trova successivo (F3)** o **Modifica** → **Trova precedente (Shift+F3)**.

3.5.3 Sostituire il testo

Per sostituire il testo avvia la barra di ricerca e sostituzione potenziata con **Modifica** → **Sostituisci**, oppure con la scorciatoia **Ctrl+R**.

In alto a sinistra nella barra c'è un pulsante con un'icona per chiudere la barra, seguito da una piccola casella combinata per inserire lo schema di ricerca. Questa casella si ricorda dei modelli usati di recente.



Puoi controllare la modalità di ricerca selezionando le opzioni **Testo semplice**, **Parole intere**, **Sequenze di escape** o **Espressione regolare** dal menu a discesa.


Se sono selezionate **Sequenze di escape** o **Espressione regolare**, l'elemento di menu **Aggiungi...** in fondo al menu contestuale delle caselle di testo sarà abilitato, e ti permetterà di aggiungere delle sequenze di escape o delle espressioni regolari per cercare o sostituire dei modelli da liste predefinite.

Usa il pulsante  oppure  per andare alla corrispondenza precedente o successiva nel documento.

Inserisci il testo da sostituire nella casella di testo indicata con **Sostituisci**, e premi il pulsante **Sostituisci** per sostituire solo il testo evidenziato; premi il pulsante **Sostituisci tutto** per sostituire il testo cercato in tutto il documento.

Puoi modificare il comportamento di ricerca e sostituzione selezionando opzioni diverse in fondo

alla barra. Selezionando  restringerai i risultati a quelli che corrispondono alle maiuscole e minuscole di ogni carattere dello schema di ricerca, mentre con  effettuerai la ricerca e la sostituzione solo all'interno della selezione attuale. Il pulsante **Trova tutti** evidenzia tutte le corrispondenze nel documento, e ne mostra il numero in una finestrella a comparsa.

Fai clic sul pulsante  sul lato destro della barra potenziata di ricerca e sostituzione per passare alla barra di ricerca incrementale.

SUGGERIMENTO

Se usi un'espressione regolare per trovare il testo da sostituire puoi utilizzare i riferimenti all'indietro per riutilizzare il testo catturato nei sottoschemi tra parentesi dell'espressione. Vedi la sezione [Espressioni regolari](#) per maggiori informazioni.

SUGGERIMENTO

Puoi usare i comandi **find**, **replace** e **ifind** (ricerca incrementale) dalla [riga di comando](#).

3.6 Usare i segnalibri

I segnalibri ti consentono di marcare alcune righe per poterle ritrovare successivamente in modo semplice.

Puoi impostare o rimuovere un segnalibro in una riga in due modi:

- Spostando il cursore di inserimento alla riga e attivando il comando **Segnalibri** → **Metti segnalibro** (**Ctrl+B**).
- Facendo clic sul bordo per le icone in corrispondenza della riga.

I segnalibri sono raggiungibili dal menu **Segnalibri**. Ognuno di essi è disponibile come voce di menu, etichettata con il numero di riga a cui è associato il segnalibro e i primi caratteri del testo di tale riga. Per spostare il cursore di inserimento all'inizio di una riga con segnalibro apri il menu e seleziona il segnalibro.

Per spostarti rapidamente tra i segnalibri o andare al segnalibro successivo o precedente, usa i comandi **Segnalibri** → **Successivo** (**Alt+Pag↓**) o **Segnalibri** → **Precedente** (**Alt+Pag↑**).

3.7 Andare a capo automaticamente

Questa funzione ti consente di formattare il testo in modo semplice: il testo andrà a capo automaticamente in modo che nessuna riga possa contenere più di un determinato numero di caratteri, a meno che non ci sia una stringa più lunga senza spazi.

Per attivarla o disattivarla attiva o disattiva la casella **Abilita a capo automatico statico** nella [pagina di modifica](#) della [finestra di configurazione](#).

Per impostare la larghezza massima di una riga (il numero massimo di caratteri) usa l'opzione **Colonna a cui andare a capo**: nella [pagina di modifica](#) della [finestra di configurazione](#).

Se attivo ha i seguenti effetti:

- Mentre digiti l'editor interrompe automaticamente la riga dopo l'ultimo carattere separatore presente in posizione precedente al raggiungimento della dimensione massima per la riga.
- In fase di caricamento di un documento l'editor manderà a capo il testo in modo analogo, facendo sì che non ci siano righe più lunghe della dimensione massima consentita se contengono dei caratteri separatori che lo consentono.

NOTA

Non c'è modo al momento di impostare l'a capo automatico per tipo di documento, e neppure di abilitarlo o disabilitarlo per ogni singolo documento. Questo problema sarà risolto in una futura versione di KatePart.

3.8 Usare il rientro automatico

Il componente editor di KatePart supporta differenti modalità di auto-rientro progettate per differenti formati di testo. Puoi scegliere tra le modalità disponibili tramite la voce di menu **Strumenti** → **Rientro**. Il modulo di auto-rientro mette a disposizione la funzione **Strumenti** → **Rientro formato** che ricalcola il rientro della riga selezionata o di quella corrente. In questo modo puoi far rientrare nuovamente il documento selezionando tutto il testo e attivando tale azione.

Tutte le modalità di rientro usano le impostazioni relative al rientro per il documento corrente.

SUGGERIMENTO

Puoi impostare tutti i tipi di variabili di configurazione, comprese quelle relative al rientro, con le [variabili dei documenti](#) e i [tipi di file](#).

MODALITÀ DI AUTO-RIENTRO DISPONIBILI

Nessuno

Questa modalità disabilita completamente il rientro.

Normale

Questa modalità si limita a mantenere il rientro simile a quello della riga precedente con contenuto diverso da spazi e separatori. Puoi combinarne l'utilizzo con quello delle azioni di rientro e di rimozione del rientro per far rientrare come meglio credi.

Stile C

Un sistema di rientro per il linguaggio C ed altri linguaggi analoghi, come C++, C#, JJava™, JavaScript e così via. Questo sistema non funziona per i linguaggi di scripting come Perl o PHP.

Haskell

Un sistema di rientro specifico per il linguaggio di programmazione funzionale Haskell.

Lilypond

Un sistema di rientro specifico per il linguaggio di notazione musicale Lilypond.

Lisp

Un sistema di rientro specifico per il linguaggio di scripting Lisp e i suoi dialetti.

Python

Un sistema di rientro specifico per il linguaggio di scripting Python.

Stile XML

Un sistema di rientro specifico i linguaggi simili a XML.

3.9 Indicatori di modifica delle righe

Gli indicatori di modifica delle righe di KatePart permettono di vedere immediatamente cosa è stato appena modificato in un file. Come impostazione predefinita le modifiche salvate vengono indicate da una barra verde a sinistra del documento, mentre quelle non salvate sono indicate in arancione.

```
// new block for empty buffer I write new stuff HERE
andNewStuffHere = 1;
if (newLineChangedAgain)
    lala ();
more = 1;
afterSaveEdited ();

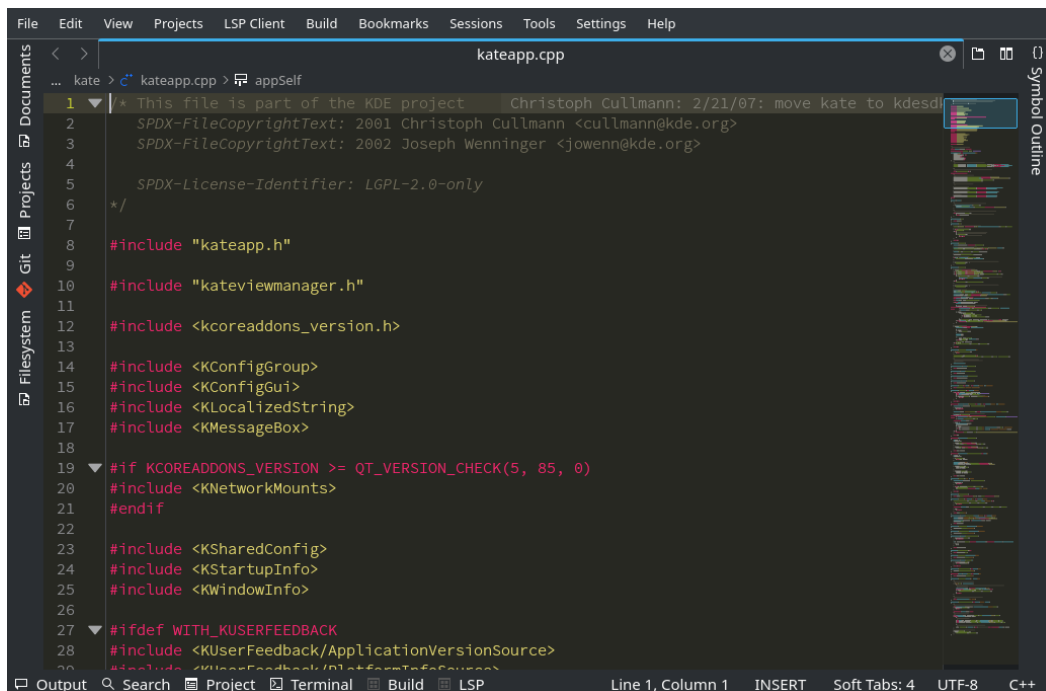
TextBlock *newBlock = new TextBlock (this, 0);
newBlock->appendLine (TextLine (new TextLineData()));
```

Gli indicatori di modifica delle righe in azione.

Puoi cambiare i colori usati nel pannello di configurazione [Caratteri e colori](#), oppure puoi disattivare del tutto questa funzionalità [nella scheda Bordi](#) del pannello di configurazione [Aspetto](#).

3.10 La mini-mappa di scorrimento

La mini-mappa di scorrimento di KatePart mostra un'anteprima dei documenti al posto della barra di scorrimento. La porzione attualmente visibile del documento viene evidenziata.



La mini-mappa di scorrimento mostra un'anteprima del codice sorgente di Kate.

Puoi abilitare o disabilitare temporaneamente la mini-mappa selezionando **Visualizza** → **Mostra mini-mappa di scorrimento** o impostarla permanentemente nella [sezione Aspetto](#) della configurazione di [KatePart](#).

3.11 Cursori multipli

Il supporto per i cursori multipli è stato introdotto nella versione 5.93 di katepart.

3.11.1 Creazione di più cursori

- Per crearli col mouse, usa **Alt** + tasto sinistro del mouse. Il modificatore è configurabile, vedi [Configurare il modificatore multi cursore](#)
- Per crearli con la tastiera, premi **Ctrl+Alt+↑** per creare un cursore sopra al primario, e **Ctrl+Alt+↓** per crearne uno sotto. Anche queste scorciatoie sono configurabili
- Per creare dei cursori da una selezione, per prima cosa seleziona del testo, poi premi **Shift+Alt+I**: ciò creerà un cursore alla fine di ogni riga nella selezione.
- Usa **Alt+J** per trovare la successiva occorrenza della parola sotto il cursore, e seleziona + per creare un cursore. Premi **Alt+K** se vuoi saltare la parola attualmente sotto al cursore: in questo modo questa verrà marcata come saltata. Se premi nuovamente **Alt+J**, ciò deselezionerà la parola corrente e passerai alla successiva.
- Usa **Ctrl+Alt+Shift+J** per cercare tutte le occorrenze della parola sotto il cursore, poi seleziona con un cursore alla fine di ogni selezione. Puoi usare **Alt+J** per passare ciclicamente tra le parole selezionate, e, come menzionato nel paragrafo precedente, **Alt+K** per deselezionare tutte le parole.

3.11.2 Lavorare con più cursori

Una volta che hai creato alcuni cursori puoi eseguire la maggior parte delle operazioni di modifica su di essi come le faresti con uno singolo. Ad esempio, digitando una lettera, questa verrà digitata per ogni cursore. Allo stesso modo puoi eseguire le trasformazioni di testo, ad es. la capitalizzazione di tutte le posizioni o le selezioni.

A volte vorrai rimuovere i cursori; per farlo, premi **Alt** + tasto sinistro del mouse su quello che vuoi rimuovere. Se vuoi solamente rimuovere i cursori nelle righe vuote c'è già un'azione pronta che lo farà per te. Per invocarla, apri la barra dei comandi usando **Ctrl+Alt+I** e cerca **Rimuovi i cursori dalle righe vuote**, poi premi **Invio**. Puoi anche configurare una scorciatoia per questa azione.

Capitolo 4

Le voci dei menu

4.1 Il menu File

File → Nuovo (Ctrl+N)

Crea un nuovo documento in una nuova finestra indipendente dell'editor.

File → Nuova finestra

Crea una nuova finestra contenente il documento corrente. Tutte le modifiche al documento fatte in una finestra saranno applicate anche nell'altra e viceversa.

File → Apri... (Ctrl+O)

Visualizza una finestra **Apri file** standard di KDE. Usa la vista dei file per selezionare il file da aprire, e fai clic su **Apri** per aprirlo.

File → Apri recenti

Questa è una scorciatoia per aprire i documenti salvati recentemente. Facendo clic su questo elemento si apre una lista a lato del menu rappresentante i file salvati più di recente. Facendo clic su uno specifico file questo si aprirà in KatePart - se il file è ancora presente nella stessa posizione.

File → Salva (Ctrl+S)

Salva il documento corrente. Se il documento è già stato precedentemente salvato verrà sovrascritto senza chiedere il consenso dell'utente. Se questa è la prima volta che si salva un nuovo documento sarà richiamata la finestra di dialogo salva come (descritta sotto).

File → Salva come... (Ctrl+Shift+S)

Permette di salvare un documento con un nuovo nome. Questo viene fatto attraverso la finestra di dialogo descritta sopra nella sezione [Apri](#) di questo file di guida.

File → Salva come con codifica

Salva un documento con un nuovo nome in una codifica diversa.

File → Salva copia come

Salva una copia del documento con un nuovo nome, e continua a modificare il documento originale.

File → Ricarica (F5)

Ricarica il file attivo da disco. Questo comando è utile se un altro programma o processo ha cambiato il file mentre era aperto in KatePart.

File → Stampa... (Ctrl+P)

Apre una semplice finestra di dialogo di stampa che permette all'utente di specificare cosa, dove e come stampare.

File → Esporta come HTML

Salva il documento attualmente aperto come file HTML, che sarà formattato usando l'attuale evidenziazione della sintassi e lo schema di colori impostato.

File → Chiudi (Ctrl+W)

Con questo comando si chiude il file attivo. Se sono presenti delle modifiche non salvate, verrà chiesto di salvare il file prima che KatePart lo chiuda.

File → Esci (Ctrl+Q)

Chiude la finestra dell'editor. Se c'è più di una istanza di KatePart in esecuzione, aperta attraverso gli elementi del menu **Nuovo** o **Nuova finestra**, queste istanze non saranno chiuse.

4.2 Il menu Modifica

Modifica → Annulla (Ctrl+Z)

Annulla l'ultima azione di modifica (inserimento di testo, copia, taglio ecc.).

NOTA

Questo può annullare molti comandi di modifica dello stesso tipo, come le digitazioni di caratteri.

Modifica → Rifai (Ctrl+Shift+Z)

Questo annullerà la modifica più recente (se c'è) fatta usando Annulla.

Modifica → Taglia (Ctrl+X)

Questo comando cancella la selezione corrente, e la mette negli appunti. Gli appunti lavorano in maniera invisibile, e forniscono un modo per trasferire dati tra le applicazioni.

Modifica → Copia (Ctrl+C)

Copia la selezione corrente negli appunti, così che possa essere incollata da qualche altra parte. Gli appunti lavorano in maniera invisibile, e forniscono un modo per trasferire dati tra le applicazioni.

Modifica → Copia come HTML

Copia la selezione come HTML, formattato usando l'attuale evidenziazione della sintassi e lo schema di colori impostato.

Modifica → Incolla (Ctrl+V)

Inserisce il primo elemento negli appunti alla posizione del cursore. Gli appunti lavorano in maniera invisibile, e forniscono un modo per trasferire dati tra le applicazioni.

NOTA

Se «Sovrascrivi la selezione» è abilitato, il testo incollato sovrascriverà la selezione, se presente.

Modifica → Incolla la selezione (Ctrl+Shift+Ins)

Questo incollerà il contenuto della [selezione del mouse](#) che era stato scelto in precedenza. Seleziona del testo col puntatore del mouse per incollarlo nel file attualmente aperto utilizzando questa voce di menu.

Modifica → Scambia con il contenuto degli appunti

Questo scambierà il testo selezionato col contenuto degli appunti.

Modifica → Incolla da cronologia degli appunti

Ciò aprirà una finestra per selezionare e incollare una voce dalla cronologia degli appunti.

Modifica → Modalità di inserimento

Passa da una modalità di editing normale ad una stile Vi, modale. La modalità Vi supporta i comandi più usati, gli spostamenti dalla modalità normale alla visuale di vim ed ha una barra di stato facoltativa per la modalità vi. Questa barra visualizza i comandi che vengono dati, i loro risultati e la modalità corrente. Il comportamento di questa modalità può essere configurato nella scheda [Modalità di inserimento Vi](#) della pagina **Modifica** nella finestra delle impostazioni di KatePart.

Modifica → Modalità di sovrascrittura (Ins)

Alterna la modalità di Inserimento/Sovrascrittura. Quando la modalità è **INS**, i caratteri vengono inseriti alla posizione del cursore. Quando la modalità è **SSC**, i caratteri digitati sostituiranno quelli correnti se il cursore è posizionato prima di qualche carattere. La barra di stato mostra lo stato corrente della Modalità di sovrascrittura: **INS** o **SSC**.



Modalità → Modalità di sola lettura

Imposta il documento corrente in Modalità di sola lettura. Ciò impedisce qualsiasi inserimento di testo e ogni modifica della formattazione del documento.

Modifica → Trova... (Ctrl+F)


Apri la barra di ricerca incrementale nella parte bassa della finestra dell'editor. Nella parte sinistra della barra c'è un pulsante con un'icona per chiuderla, seguita da una piccola casella di testo per inserire la chiave di ricerca.


Quando si inizia ad inserire la chiave di ricerca, la ricerca inizia immediatamente. Se c'è una corrispondenza nel testo, questa viene evidenziata e lo sfondo della casella di testo diventa verde chiaro. Se la ricerca non dà risultati, lo sfondo della casella di testo diventa rosso chiaro.

Si usa il pulsante  oppure  per andare alla corrispondenza precedente o successiva nel documento.

Le corrispondenze nel documento sono evidenziate anche quando chiudi la barra di ricerca; premi il tasto **Esc** per togliere questa evidenziazione.

Permette di limitare la ricerca alle corrispondenze con le stesse maiuscole nello schema di

ricerca. Selezionando  si limiterà la ricerca ai risultati con le maiuscole e minuscole uguali a quelle specificate nella ricerca.

Fare clic sul pulsante  nella parte destra della barra di ricerca incrementale per usare la barra avanzata di trova e sostituisci.

Modifica → Trova varianti → Trova successivo (F3)

Ripete l'ultima ricerca, se c'è, senza richiamare la barra di ricerca incrementale, e cercando in avanti nel documento, a partire dalla posizione del puntatore.

Modifica → Trova varianti → Trova precedente (Shift+F3)

Ripete l'ultima ricerca, se c'è, senza richiamare la barra di ricerca incrementale e cercando all'indietro invece che in avanti nel documento.

Edit → Trova varianti → Trova selezionato (Ctrl+H)

Trova la prossima occorrenza del testo selezionato.

Modifica → Trova varianti → Trova selezionato all'indietro (Ctrl+Shift+H)



Trova l'occorrenza precedente del testo selezionato.

Modifica → Sostituisci... (Ctrl+R)



Questo comando apre la barra avanzata di trova e sostituisci. Nella parte a sinistra in alto della barra c'è un pulsante con un'icona per chiuderla, seguita da una piccola casella di testo per inserire la chiave di ricerca.

Si può modificare la modalità di ricerca marcando **Testo semplice**, **Parole intere**, **Sequenze di escape** o **Espressioni regolari** dal menu a discesa.

Se sono attivi **Sequenze di escape** o **Espressione regolare**, l'elemento **Aggiungi...** in basso nel menu contestuale delle caselle di testo è abilitato e permette di aggiungere sequenze di escape o elementi di espressioni regolari alle chiavi di ricerca, da liste predefinite.

Si usa il pulsante  oppure  per andare alla corrispondenza precedente o successiva nel documento.

Inserire il testo con cui sostituire nella casella di testo **Sostituisci** e fare clic sul pulsante **Sostituisci** per sostituire solo il testo evidenziato o sul pulsante **Sostituisci tutti** per sostituire il testo in tutto il documento.

Si può modificare il comportamento di trova e sostituisci cambiando le opzioni selezionate nella parte destra della barra. Marcando  si fanno corrispondere solo caratteri maiuscoli a caratteri maiuscoli (e viceversa) nella ricerca.  cercherà corrispondenze solo nella selezione corrente. Il pulsante **Trova tutto** evidenzia tutte le corrispondenze nel documento e ne visualizza il numero in una piccolo popup.

Fare clic sul pulsante  nella parte destra della barra avanzata di trova e sostituisci per usare la barra di ricerca incrementale.

4.3 Il menu Selezione

Selezione → Seleziona tutto (Ctrl+A)

Seleziona l'intero documento. Può essere molto utile per copiare l'intero file in un'altra applicazione.

Selezione → Deseleziona (Ctrl+Shift+A)

Deseleziona il testo selezionato nell'editor, se è presente.

Selezione → Modalità di selezione a blocchi (Ctrl+Shift+B)

Alterna la modalità di selezione. Quando la modalità di selezione è **BLOCCO** la barra di stato contiene la stringa **[BLOCCO]** e si possono fare delle selezioni verticali, ad es. selezionare le colonne dalla 5 alla 10 nelle righe dalla 9 alla 15.

Selezione → Commento (Ctrl+D)

Aggiunge uno spazio all'inizio della riga dove è posizionato il cursore o all'inizio delle righe selezionate.

Selezione → Decomenta (Ctrl+Shift+D)

Rimuove uno spazio (se esiste) dall'inizio della riga dove è posizionato il cursore o dall'inizio delle righe selezionate.

Selezione → Unisci righe (Ctrl+J)

Unisce le righe selezionate, o la riga corrente e quella successiva, con un carattere di spazio come separatore. Gli spazi bianchi iniziali o finali nelle righe unite saranno rimossi negli estremi coinvolti.

Selezione → Iniziali maiuscole (Ctrl+Alt+U)

Trasforma le iniziali in maiuscole nel testo selezionato o nella parola corrente.

Selezione → Maiuscolo (Ctrl+U)

Trasforma il testo selezionato o la lettera che segue il cursore in maiuscolo.

Selezione → Minuscolo (Ctrl+Shift+U)

Trasforma il testo selezionato o la lettera che segue il cursore in minuscolo.

Selezione → Pulisci rientro

Pulisce il rientro per la selezione corrente o per la riga su cui si trova al momento il cursore. Pulire il rientro garantisce che tutto il testo selezionato segua la modalità di rientro scelta.

Selezione → Rientro formato

Provoca un riallineamento della riga corrente o delle righe selezionate utilizzando la modalità di rientro e le impostazioni di rientro del documento.

Selezione → Allinea a...

Questo comando allinea, alla colonna data da un'espressione regolare che dovrai inserire, le righe nel blocco selezionato o nell'intero documento.

Per impostazione predefinita, se inserisci un modello vuoto l'allineamento sarà col primo carattere non di spazio.

Se il modello ha una cattura, il rientro sarà con la prima corrispondenza catturata.

Esempi:

Con «-» verrà inserito uno spazio prima del primo «-» di ogni riga per allinearle nella stessa colonna.

Con «alignon :\\s+(.)» inserirà degli spazi prima del primo carattere non di spazio che ricorre dopo i due punti per allinearle nella stessa colonna.

Selezione → A capo automatico

Applica l'a capo statico sull'intero documento. Ciò significa che verrà creata automaticamente una nuova riga di testo quando la riga corrente supererà la lunghezza specificata dall'opzione **Torna a capo dopo** nella scheda Modifica nel menu **Impostazioni → Configura editor...**

Selezione → Aggiungi un accento circonflesso sopra il cursore (Ctrl+Alt+↑)

Aggiunge un accento circonflesso sopra il cursore corrente. L'accento circonflesso viene messo nella riga immediatamente sopra e nella stessa colonna del cursore corrente.

Selezione → Aggiungi un accento circonflesso sotto il cursore (Ctrl+Alt+↓)

Aggiunge un accento circonflesso sotto il cursore corrente. L'accento circonflesso viene messo nella riga immediatamente sopra e nella stessa colonna del cursore corrente.

Selezione → Aggiungi cursori di fine riga (Alt+Shift+I)

Aggiunge un cursore ad ogni riga attualmente selezionata.

Selezione → Trova e seleziona la prossima occorrenza (Alt+J)

Trova la successiva occorrenza della parola attualmente sotto il cursore, la seleziona e aggiunge un cursore.

Selezione → Trova e seleziona tutte le occorrenze (Ctrl+Alt+Shift+J)

Trova tutte le occorrenza della parola attualmente sotto il cursore, le seleziona e aggiunge un cursore per ciascuna di esse.

4.4 Il menu Visualizza

Visualizza → Ingrandisci i caratteri (Ctrl++)

Aumenta la dimensione del carattere.

Visualizza → Rimpicciolisci i caratteri (Ctrl+-)

Diminuisce la dimensione del carattere.

Visualizza → A capo automatico → Ritorno a capo dinamico (F10)

Attiva o disattiva l'andata a capo automatica dinamica della vista attuale. L'andata a capo dinamica rende visibile tutto il testo di una vista senza scorrere orizzontalmente, facendo mostrare una riga di testo effettivo su una o più righe quando necessario.

Visualizza → A capo automatico → Indicatori di a capo dinamico

Scegli quando e come debbano essere visualizzati gli indicatori di andata a capo dinamica. Ciò è disponibile solo se l'opzione **Ritorno a capo dinamico** è attivata.

Visualizza → A capo automatico → Mostra indicatore di ritorno a capo statico

Se questa opzione è attivata, verrà disegnata una barra verticale sulla colonna di andata a capo definita in **Impostazioni → Configura editor...** nella scheda Modifica. Attenzione che l'indicatore di andata a capo sarà disegnato solo se si utilizza un carattere a spaziatura fissa.

Visualizza → Bordi → Mostra bordo delle icone (F6)

Questa è una voce di commutazione. Renderà visibile o meno il bordo per le icone sul lato sinistro dell'editor attivo. Il bordo dell'icona indica le posizioni delle righe segnate nell'editor.

Visualizza → Bordi → Mostra i numeri di riga (F11)

Questo è un elemento alternato. Attivandolo verrà visualizzato un pannello sul bordo sinistro dell'editor attivo, per visualizzare i numeri di riga del documento, e viceversa.

Visualizza → Bordi → Mostra i segni nella barra di scorrimento

Se questa opzione è attivata, la vista mostrerà indicatori sulla barra di scorrimento verticale. I segni equivalgono a quelli sul [bordo delle icone](#).

Visualizza → Bordi → Mostra minimappa di scorrimento

Questo sostituirà la barra di scorrimento con una visualizzazione del documento attuale. Per maggiori informazioni sulla minimappa di scorrimento, vedi Sezione [3.10](#).

Visualizza → Raggruppamento del codice

Queste opzioni si riferiscono al [raggruppamento del codice](#):

Visualizza → Bordi → Raggruppamento del codice → Mostra segni di raggruppamento

Attiva e disattiva la visualizzazione degli indicatori di raggruppamento a sinistra della vista.

Visualizza → Raggruppamento del codice → Contrai il nodo attuale

Contrae la regione contenente il cursore.

Visualizza → Raggruppamento del codice → Espandi il nodo attuale

Espande la regione contenente il cursore.

Visualizza → Raggruppamento del codice → Contrai nodi di massimo livello (Ctrl+Shift+-)

Contrae tutte le regioni di massimo livello nel documento. Fai clic sul triangolo che punta a destra per espandere tutte le regioni di massimo livello.

Visualizza → Raggruppamento del codice → Espandi nodi di massimo livello. (Ctrl+Shift++)

Espande tutte le regioni di livelli superiori nel documento.

Mostra spazi non stampabili

Mostra/nasconde i riquadri che delimitano gli spazi non stampabili.

4.5 Il menu Vai

Vai → Vai alla riga... (Ctrl+G)

Apri la barra vai alla riga nella parte bassa della finestra, che serve per far saltare il cursore ad una particolare riga (specificata dal numero) nel documento. Il numero della riga può essere inserito direttamente nella casella di testo o graficamente, facendo clic sulle frecce direzionali su o giù a lato della casella di testo. La piccola freccia su incrementerà il numero della riga e la freccia giù lo decreterà. Si può chiudere la barra facendo clic sul pulsante con un'icona nella parte sinistra.

Vai → Vai alla riga di modifica precedente (Ctrl+E)

Questa azione salta alla precedente riga di modifica nella [configurazione a più cursori](#).

Vai → Vai alla riga di modifica successiva (Ctrl+Shift+E)

Questa azione salta alla successiva riga di modifica nella [configurazione a più cursori](#).

Vai → Vai alla riga modificata precedente

Le righe che sono cambiate dall'apertura del file sono chiamate righe modificate. Questa azione salta alla riga modificata precedente.

Vai → Vai alla riga modificata successiva

Le righe che sono cambiate dall'apertura del file sono chiamate righe modificate. Questa azione salta alla riga modificata successiva.

Vai → Vai alla parentesi corrispondente (Ctrl+6)

Sposta il cursore alla parentesi di apertura o di chiusura associata.

Vai → Seleziona fino alla parentesi corrispondente (Ctrl+Shift+6)

Seleziona il testo tra le parentesi di apertura e di chiusura associate.

Vai → Segnalibri (Ctrl+Shift+6)

Sotto le voci qui descritte sarà disponibile una voce per ciascun segnalibro nel documento attivo. Il testo sarà composto dalle prime parole della riga indicata. Scegli una voce per spostare il cursore all'inizio della riga corrispondente. L'editor scorrerà il testo di quanto è necessario per mostrare la riga.

Vai → Segnalibri → Imposta segnalibro (Ctrl+B)

Imposta o rimuove un segnalibro dalla riga corrente del documento attuale. (se è già presente verrà rimosso, altrimenti ne verrà impostato uno).

Vai → Segnalibri → Togli tutti i segnalibri

Questo comando rimuoverà tutti i segnalibri dal documento, così come la lista dei segnalibri che si trova sotto questo elemento del menu.

Vai → Segnalibri → Precedente (Alt+Pag↑)

Sposta il cursore all'inizio della prima riga sopra quella attuale che ha un segnalibro. Il testo dell'elemento nel menu conterrà il numero di riga e il primo pezzo del testo nella riga. Questa voce è disponibile solo quando esiste un segnalibro in una riga sopra al cursore.

Vai → Segnalibri → Successivo (Alt+Pag↓)

Sposta il cursore all'inizio della prossima riga che ha un segnalibro. La voce nel menu conterrà il numero della riga e il primo pezzo del testo nella riga. Questa voce è disponibile quando esiste un segnalibro in una riga sotto al cursore.

4.6 Il menu Strumenti

Strumenti → Modalità

Sceglie lo schema del tipo di file preferito per il documento attuale. Questo avrà priorità sullo schema scelto a livello globale in **Impostazioni** → **Configura editor...** nella scheda Modi e tipi di file, unicamente per il documento corrente.

Strumenti → Evidenziazione

Sceglie lo schema di evidenziazione preferito per il documento attivo. Questo ha priorità rispetto alla modalità di evidenziazione globale impostata in **Impostazioni** → **Configura editor...** unicamente per il documento attuale.

Strumenti → Rientro

Sceglie lo stile di rientro desiderato per il documento attivo. Questo ha priorità rispetto alla modalità di rientro globale impostata in **Impostazioni** → **Configura editor...** unicamente per il documento attuale.

Strumenti → Codifica

Si può sovrascrivere l'impostazione di codifica predefinita in **Settings** → **Configura editor...** nella pagina **Apri e salva** per impostare una differente codifica per il documento corrente. La codifica impostata qui sarà valida unicamente per il documento corrente.

Strumenti → Aggiungi indicatore dell'ordine dei byte (BOM)

Attivando questa azione puoi aggiungere esplicitamente un indicatore dell'ordine dei byte per i documenti codificati con Unicode. L'indicatore dell'ordine dei byte (BOM) è un carattere Unicode usato per indicare appunto l'ordine dei byte (in gergo inglese, *endianness*) di un file o flusso di testo; per maggiori informazioni vedi su [Wikipedia](#).

Strumenti → Fine riga

Sceglie la modalità di fine riga preferita per il documento attivo. Questa avrà priorità rispetto alla modalità di fine riga globale impostata in **Settings** → **Configura editor...** unicamente per il documento corrente.

Strumenti → Script

Questo sottomenu contiene un elenco di tutte le azioni di script. L'elenco può essere facilmente modificato [scrivendo i tuoi script](#). In questo modo, si può estendere KatePart con strumenti definiti dall'utente.

Strumenti → Script → Navigazione

Strumenti → Script → Navigazione → Sposta il cursore al rientro corrispondente precedente (Alt+Shift+↑)

Sposta il cursore alla prima riga sopra l'attuale che sia rientrata allo stesso livello dell'attuale.

Strumenti → Script → Navigazione → Sposta il cursore al rientro corrispondente successivo (Alt+Shift+↓)

Sposta il cursore alla prima riga sotto l'attuale che sia rientrata allo stesso livello dell'attuale.

Strumenti → Script → Modifica

Strumenti → Script → Modifica → Ordina il testo selezionato

Ordina il testo selezionato o l'intero documento in ordine crescente.

Strumenti → Script → Modifica → Sposta righe in basso (Ctrl+Shift+↓)

Sposta in basso le righe selezionate.

Strumenti → Script → Modifica → Sposta righe in alto (Ctrl+Shift+↑)

Sposta in alto le righe selezionate.

Strumenti → Script → Modifica → Duplica le righe selezionate in basso (Ctrl+Alt+↓)
Duplica le righe selezionate in basso.

Strumenti → Script → Modifica → Duplica le righe selezionate in alto (Ctrl+Alt+↑)
Duplica le righe selezionate in alto.

Strumenti → Script → Modifica → Codifica il testo selezionato come URI
Codifica il testo selezionato in modo che possa essere usato come parte di una stringa di interrogazione in un URL, sostituendo la selezione con il testo codificato.

Strumenti → Script → Modifica → Decodifica il testo selezionato come URI
Se viene selezionata parte della stringa di interrogazione di un URL, questo la decodificherà e sostituirà la selezione con il testo originale.

Strumenti → Script → Emmet

Strumenti → Script → Emmet → Espandi abbreviazione
Converte il testo selezionato in una coppia di tag HTML o XML di apertura e chiusura. Per esempio, se è selezionato **div**, questo elemento lo sostituirà con `<div></div>`.

Strumenti → Script → Emmet → Racchiudi con tag
Racchiudi il testo selezionato con il tag fornito sulla [riga di comando](#).

Strumenti → Script → Emmet → Sposta il cursore al tag corrispondente
Se il cursore è all'interno di un tag di apertura HTML/XML, questo elemento lo sposterà al tag di chiusura. Se il cursore è all'interno del tag di chiusura, lo sposterà invece a quello di apertura.

Strumenti → Script → Emmet → Seleziona i contenuti dei tag HTML/XML senza tag
Quando il cursore è all'interno di una coppia di tag HTML/XML, questo elemento cambierà la selezione per includere i contenuti di quei tag HTML/XML, senza selezionare i tag stessi.

Strumenti → Script → Emmet → Seleziona i contenuti dei tag HTML/XML con tag
Quando il cursore è all'interno di una coppia di tag HTML/XML, questo elemento cambierà la selezione per includere i contenuti di quei tag HTML/XML, inclusi i tag stessi.

Strumenti → Script → Emmet → Commenta/decommenta
Se la porzione selezionata non è un commento, questo elemento racchiuderà quella porzione in un commento HTML/XML (cioè `<!-- testo selezionato -->`). Se la porzione selezionata è un commento, i tag del commento verranno invece rimossi.

Strumenti → Script → Emmet → Elimina tag sotto il cursore
Se il cursore è attualmente dentro a un tag HTML/XML, questo elemento eliminerà l'intero tag.

Strumenti → Script → Emmet → Riduci numero di 1
Questo elemento sottrarrà 1 dal testo attualmente selezionato, se è un numero. Per esempio, se è selezionato **5**, diventerà 4.

Strumenti → Script → Emmet → Riduci numero di 10
Questo elemento sottrarrà 10 dal testo attualmente selezionato, se è un numero. Per esempio, se è selezionato **15**, diventerà 5.

Strumenti → Script → Emmet → Riduci numero di 0,1
Questo elemento sottrarrà 0,1 dal testo attualmente selezionato, se è un numero. Per esempio, se è selezionato **4, 5**, diventerà 4, 4.

Strumenti → Script → Emmet → Aumenta numero di 1
Questo elemento aggiungerà 1 al testo attualmente selezionato, se è un numero. Per esempio, se è selezionato **5**, diventerà 6.

Strumenti → Script → Emmet → Aumenta numero di 10
Questo elemento aggiungerà 10 al testo attualmente selezionato, se è un numero. Per esempio, se è selezionato **5**, diventerà 15.

Strumenti → Script → Emmet → Aumenta numero di 0,1
Questo elemento aggiungerà 0,1 al testo attualmente selezionato, se è un numero. Per esempio, se è selezionato **4, 5**, diventerà 4, 6.

Strumenti → Passa alla riga di comando (F7)

Visualizza la riga di comando di KatePart alla base della finestra. Nella riga di comando, digita **help** per ottenere aiuto e **help list** per ottenere una lista di comandi. Per maggiori informazioni sulla riga di comando, vedi [la riga di comando del componente editor](#).

Strumenti → Invoca completamento del codice (Ctrl+Spazio)

Invoca manualmente il completamento del codice, spesso usando una scorciatoia associata a questa azione.

Strumenti → Completamento delle parole

Riutilizza parola in basso (Ctrl+9) e **Riutilizza parola in alto** (Ctrl+8) completano il testo digitato cercando parole simili in avanti o all'indietro dal cursore. **Completamento da shell** fa comparire un riquadro di completamento con le voci corrispondenti.

Strumenti → Ortografia → Controllo ortografico automatico (Ctrl+Shift+O)

Quando **Controllo ortografico automatico** è attivo, gli errori di ortografia rilevati vengono immediatamente sottolineati nel documento.

Strumenti → Ortografia → Ortografia...

Lancia il programma di controllo ortografico - un programma dedicato ad aiutare l'utente a trovare e correggere qualsiasi errore di scrittura. Facendo clic su questa voce comincerà il controllo e verrà aperta la finestra di dialogo del correttore, attraverso la quale l'utente può controllare il processo. Ci sono quattro impostazioni allineate verticalmente al centro della finestra con alla loro sinistra l'etichetta corrispondente. A partire dall'alto abbiamo:

Parola sconosciuta:

Qui, il controllo ortografia indica la parola sotto esame. Questo accade quando si incontra una parola non contenuta nel dizionario - un file che contiene una lista di parole corrette ortograficamente che vengono confrontate con tutte quelle presenti nell'editor.

Sostituisci con:

Se il correttore ha qualche parola simile nel suo dizionario, la prima viene inclusa qui. L'utente può accettare il suggerimento, scrivere la propria correzione, o scegliere un differente suggerimento dalla prossima casella.

Lingua:

Se sono installati diversi dizionari, qui si può scegliere il dizionario/lingua da usare.

Sul lato destro della finestra di dialogo ci sono sei pulsanti che permettono all'utente di controllare il processo di controllo ortografico. Questi sono:

Aggiungi al dizionario

Premendo questo pulsante si aggiungerà la **Parola sconosciuta** nel dizionario del correttore. Questo significa che in futuro il correttore considererà questa parola ortograficamente corretta.

Suggerisci

Il correttore può elencare qui alcune possibili sostituzioni per la parola presa in considerazione. Facendo clic su un suggerimento, questo va a sostituire la parola nella casella **Sostituisci con**, in alto.

Sostituisci

Questo pulsante ha il compito di sostituire la parola sotto esame nel documento con la parola nella casella **Sostituisci con**.

Sostituisci tutto

Questo pulsante fa sì che il controllo rimpiazzhi non solo la corrente **Parola sconosciuta** ma automaticamente faccia la stessa sostituzione per ogni altra occorrenza di questa **Parola sconosciuta** nel documento.

Ignora

Attivando questo pulsante il controllo proseguirà oltre senza applicare cambiamenti.

Ignora sempre

Questo pulsante comunica al controllo di ignorare questa **Parola sconosciuta** ed andare oltre senza considerare altre eventuali occorrenze della stessa parola.

NOTA

Questo è applicato solo al controllo corrente. Se il controllo sarà riavviato in un secondo momento, si fermerà su questa stessa parola.

Altri tre pulsanti sono posizionati lungo la base della finestra del controllo ortografico. Essi sono:

Aiuto

Apri il sistema di guida di KDE con la pagina di aiuto per questa finestra.

Finito

Questo pulsante ferma il processo di correzione e riporta al documento.

Annulla

Questo pulsante annulla il processo di controllo ortografico, annulla tutte le modifiche e fa ritornare al proprio documento.

Strumenti → Ortografia → Ortografia (dal cursore)...

Ciò avvierà il programma di correzione ortografica iniziando dalla posizione corrente del cursore invece che dall'inizio del documento.

Strumenti → Ortografia → Ortografia della selezione...

Esegue il controllo ortografico sulla selezione corrente.

Strumenti → Ortografia → Cambia dizionario

Visualizza un menu a discesa nella parte bassa della finestra dell'editor, con i dizionari disponibili per la correzione ortografica. Permette di cambiare il dizionario in modo veloce, ad esempio per la correzione ortografica di documenti in più lingue.

4.7 I menu Impostazioni e Aiuto

Impostazioni → Tema di colori dell'editor

Questo menu elenca gli schemi di colori disponibili. Qui puoi cambiare lo schema per l'attuale visualizzazione: per cambiare quello predefinito usa la pagina [Caratteri & Colori](#) nella finestra di dialogo di configurazione.

Nel menu di KatePart compaiono due comuni voci dei menu di KDE, **Impostazioni** e **Aiuto**. Per maggiori informazioni leggi le sezioni [Menu impostazioni](#) e [Menu di aiuto](#) nei Fondamentali di KDE.

Capitolo 5

Strumenti avanzati di modifica

Anders Lund
Dominik Haumann
Traduzione della documentazione.: Nicola Ruggero
Aggiornamento per Kate 2.5.6: Luciano Montanaro

5.1 Commenta/Decommenta

I comandi **Commenta** e **Decommenta**, disponibili nel menu **Strumenti**, permettono di aggiungere o di rimuovere gli indicatori di commento alla selezione, o alla riga attuale del testo se non hai selezionato niente, a patto che i commenti facciano parte del formato di testo del documento.

Le regole su come è applicato il commento sono definite in base alla sintassi del testo che stai scrivendo, quindi se non hai selezionato una modalità di evidenziazione non potrai usare **Commenta/Decommenta**.

Alcuni formati usano indicatori di commento per righe singole, altri usano indicatori per più righe consecutive di commento, ed altri ancora entrambi i sistemi. Se i marcatori multiriga non sono disponibili, non è possibile commentare una selezione che non comprenda completamente l'ultima riga.

Se sono disponibili i marcatori per righe singole è preferibile usarli ogni volta che è possibile, perché ciò evita i problemi con i commenti nidificati.

Quando rimuovi gli indicatori di commento devi fare attenzione a non selezionare del testo non facente parte del commento. Quando rimuovi i commenti su più righe da una selezione, gli spazi vuoti al di fuori degli indicatori di commento vengono ignorati.

Per inserire gli indicatori di commento, usa il comando **Strumenti** → **Commenta** oppure la relativa scorciatoia da tastiera; quella predefinita è **Ctrl+#**.

Per rimuovere gli indicatori di commento, usa il comando **Strumenti** → **Decommenta** oppure la relativa scorciatoia da tastiera; quella predefinita è **Ctrl+Shift+#**.

5.2 La riga di comando per il componente dell'editor

Il componente editor di KatePart ha una riga di comando interna che ti permette di effettuare varie azioni da una GUI minima. La riga di comando è una casella di testo in fondo all'area di modifica; per mostrarla, seleziona **Visualizza** → **Passa alla riga di comando** o usa la scorciatoia (**F7**). L'editor fornisce l'insieme di comandi documentati di seguito, inoltre con le estensioni vengono forniti dei comandi aggiuntivi.

Per eseguire un comando, scrivilo e premi il tasto Invio: la riga di comando indicherà se ha avuto successo, ed eventualmente mostrerà un messaggio. Se hai attivato la riga di comando premendo **F7**, essa verrà nascosta automaticamente dopo qualche secondo. Per pulire la riga ed inserire un nuovo comando, premi di nuovo **F7**.

La riga di comando ha un sistema di aiuto integrato; per iniziare usa il comando **help**. Per vedere un elenco di tutti i comandi disponibili, usa **help list**; per vedere l'aiuto di un comando specifico, usa **help comando**.

La riga di comando si ricorda la cronologia dei comandi, per cui puoi riutilizzare i comandi che hai già inserito. Per scorrere la cronologia, usa i tasti \uparrow e \downarrow . Quando vengono mostrati i comandi della cronologia, la parte degli argomenti del comando è selezionata, in modo da permetterti di sovrascrivere semplicemente gli argomenti.

5.2.1 I comandi standard della riga di comando

TIPI DI ARGOMENTO

BOOLEAN

Questo tipo è usato nei comandi che devono attivare o disattivare una caratteristica. I valori ammessi sono **on**, **off**, **true**, **false**, **1** o **0**.

INTEGER

Un numero intero.

STRING

Una stringa, delimitata da virgolette singole (') o doppie (") se contiene spazi.

5.2.1.1 Comandi per la configurazione dell'editor

Questi comandi sono forniti dal componente editor, e permettono di configurare solo il documento attivo e la vista. Sono utili se vuoi usare impostazioni diverse da quelle predefinite, ad esempio per i rientri.

set-tab-width INTEGER *ampiezza*

Imposta l'ampiezza della tabulazione al numero **ampiezza**.

set-indent-width INTEGER *ampiezza*

Imposta l'ampiezza del rientro al numero **ampiezza**. È usata solo se fai rientrare con gli spazi.

set-word-wrap-column INTEGER *ampiezza*

Imposta l'ampiezza della riga al numero **ampiezza**. È usata se il testo deve andare a capo automaticamente.

set-icon-border BOOLEAN *abilitazione*

Imposta la visibilità del bordo per le icone.

set-folding-markers BOOLEAN *abilitazione*

Imposta la visibilità del pannello per gli indicatori di raggruppamento.

set-line-numbers BOOLEAN *abilitazione*

Imposta la visibilità del pannello per i numeri di riga.

set-replace-tabs BOOLEAN *abilitazione*

Se è abilitata, le tabulazioni sono sostituite con spazi durante l'inserimento.

set-remove-trailing-space BOOLEAN **abilitazione**

Se è abilitata, gli spazi in fin di riga vengono rimossi quando il cursore lascia una riga.

set-show-tabs BOOLEAN **abilitazione**

Se è abilitata, le tabulazioni e gli spazi in fin di riga saranno evidenziati con dei puntini.

set-show-indent BOOLEAN **abilitazione**

Se è abilitata, il rientro sarà evidenziato con una linea punteggiata verticale.

set-indent-spaces BOOLEAN **abilitazione**

Se è abilitata, l'editor farà rientrare il testo di `indent-width` spazi ad ogni livello di rientro, invece che di un carattere di tabulazione.

set-mixed-indent BOOLEAN **abilitazione**

Se abilitato, KatePart userà una miscela di tabulazioni e spazi per i rientri delle righe. Ciascun livello di rientro sarà largo `indent-width`, e più livelli di rientro saranno ottimizzati usando il maggior numero possibile di tabulazioni.

Quando viene eseguito, questo comando abilita inoltre i rientri con gli spazi, e, se l'ampiezza dei rientri non è specificata, sarà impostata alla metà di `tab-width` del documento al momento dell'esecuzione.

set-word-wrap BOOLEAN **abilitazione**

Abilita o disabilita l'a capo automatico dinamico a seconda del parametro **abilitazione**.

set-replace-tabs-save BOOLEAN **abilitazione**

Quando è attiva questa opzione, le tabulazioni verranno sostituite con spazi durante il salvataggio del documento.

set-remove-trailing-space-save BOOLEAN **abilitazione**

Quando è attiva questa opzione, gli spazi alla fine delle righe saranno scartati durante il salvataggio del documento.

set-indent-mode STRING **nome**

Imposta la modalità di autorientro a **nome**. Se **nome** è sconosciuto, la modalità sarà impostata a **none**. Le modalità valide sono **none**, **normal**, **cstyle**, **haskell**, **lilypond**, **lisp**, **python**, **ruby**, e **xml**.

set-auto-indent BOOLEAN **script**

Abilita o disabilita il rientro automatico.

set-highlight STRING **evidenziazione**

Imposta il sistema di evidenziazione della sintassi per il documento. L'argomento deve essere un nome di evidenziazione valido, così come è indicato nel menu **Strumenti** → **Evidenziazione**. Questo comando fornisce un elenco per il completamento automatico dell'argomento.

reload-scripts

Ricarica tutti gli [script in JavaScript](#) usati da Kate, inclusi i rientratori e gli script dalla riga di comando.

set-mode STRING **modalità**

Scegli lo schema di tipo di file per il documento attuale.

nn[oremap] STRINGA **originale** STRINGA **mappata**

Mappa la sequenza di tasti **originale** nella **mappata**.

5.2.1.2 Comandi di modifica

Questi comandi servono a modificare il documento attuale.

indent

Fa rientrare le righe selezionate o la riga attuale.

unindent

Riduce il rientro delle righe selezionate o di quella attuale.

cleanindent

Riordina il rientro delle righe selezionate o della riga attuale secondo le impostazioni per il rientro del documento.

comment

Inserisce indicatori di commento per rendere la selezione o le righe selezionate o la riga attuale un commento secondo il formato di testo così com'è definito dalle regole di evidenziazione della sintassi del documento.

uncomment

Rimuove gli indicatori di commento dalla selezione o dalle righe selezionate o dalla riga attuale secondo il formato di testo così com'è definito dalle regole di evidenziazione della sintassi del documento.

kill-line

Elimina la riga attuale.

replace STRINGA modello STRINGA sostituzione

Sostituisce il testo corrispondente a **modello** con **sostituzione**. Se vuoi includere spazi nel **modello**, devi mettere le virgolette single o doppie attorno sia al modello, sia alla sostituzione. Se gli argomenti non sono tra virgolette, la prima parola è usata come **modello** ed il resto come **sostituzione**. Se la **sostituzione** è vuota, vengono rimosse le corrispondenze con il **modello**.

Puoi impostare dei modificatori per configurare la ricerca aggiungendo un due punti, seguiti da una o più lettere rappresentanti la configurazione, data nella forma **replace:opzioni modello sostituzione**. Le opzioni disponibili sono:

b

Cerca all'indietro.

c

Cerca dalla posizione del cursore.

e

Cerca solo all'interno della selezione.

r

Esegue la ricerca con un'espressione regolare. Se è impostata, puoi usare **\N**, dove N è un numero che rappresenta le catture nella stringa di sostituzione.

s

Esegue la ricerca facendo distinzione fra maiuscole e minuscole.

p

Chiede il permesso prima di effettuare la sostituzione.

w

Fa in modo che la ricerca consideri solo parole intere.

date STRINGA formato

Inserisce la stringa dell'ora/data come è specificato in **formato**, o nel formato 'yyyy-MM-dd hh:mm:ss' se non ne è specificato nessuno. Le seguenti traduzioni sono effettuate nell'interpretazione di **formato**:

d	Il giorno in forma numerica, senza zeri iniziali (1–31).
dd	Il giorno in forma numerica, con uno zero iniziale di allineamento (01–31).
ddd	Il nome del giorno abbreviato in forma locale (ad es.: «lun» ... «dom»).
dddd	Il nome del giorno completo in forma locale (ad es. «lunedì» ... «domenica»).
M	Il mese in forma numerica, senza zeri iniziali (1–12).
MM	Il mese in forma numerica, con uno zero iniziale se necessario (01–12).
MMMM	Il nome del mese per esteso in forma locale (ad es. «Gennaio» ... «Dicembre»).
MMM	Il nome del mese abbreviato in forma locale (ad es. «gen» ... «dic»).
yy	Le ultime due cifre dell'anno (00–99).
yyyy	L'anno come numero di quattro cifre (1752–8000).
h	L'ora senza zero iniziali (0–23 o 1–12 se in modalità AM/PM).
hh	L'ora con zero iniziali (00–23 o 01–12 se in modalità AM/PM).
m	I minuti senza zero iniziali (0–59).
mm	I minuti con zero iniziali (00–59).
s	I secondi senza zero iniziali (0–59).
ss	I secondi con zero iniziali (00–59).
z	I millisecondi senza zeri iniziali (0–999).
zzz	I millisecondi con zeri iniziali (000–999).
AP	Usa modalità AM/PM. AP sarà sostituito da "AM" per le ore antimeridiane o "PM" per quelle pomeridiane.
ap	Usa modalità am/pm. ap sarà sostituito da "am" per le ore antimeridiane o "pm" per quelle pomeridiane.

char STRINGA identificativo

Questo comando permette di inserire caratteri indicando il loro identificativo numerico, in decimale, ottale o esadecimale. Per usarlo, apri la finestra dei comandi di modifica e digita **char:** [numero] nel campo di inserimento, quindi fai clic su **OK**.

Example 5.1 Esempi con il comando char

Inserisci: **char:234**

Ottieni: ê

Inserisci: **char:0x1234**

Ottieni: jué

s///[ig] %s///[ig]

Questo comando fa una ricerca/sostituzione nella riga di testo corrente o in tutto il file (%s///) in maniera del tutto simile a sed.

In breve, la ricerca viene effettuata in base a un *modello di ricerca*, a un'espressione regolare compresa tra la prima e la seconda barra. Quando viene trovata una corrispondenza, la parte che corrisponde viene sostituita con l'espressione compresa tra la seconda e l'ultima barra. Le parentesi nel modello di ricerca creano dei *referimenti all'indietro*, vale a dire che il comando ricorda quali parti della stringa di ricerca tra parentesi hanno trovato corrispondenza; queste stringhe possono essere riutilizzate per sostituire dei modelli riferiti a \1 per il primo gruppo di parentesi, \2 per il secondo e così via.

Per cercare precisamente il simbolo di parentesi (o), devi *evitare* il problema usando la barra rovescia in questo modo: \ (\)

Se metti una **i** alla fine dell'espressione, la corrispondenza avverrà senza tener far distinzione fra maiuscole o minuscole. Se metti una **g**, verranno sostituite tutte le corrispondenze del modello, altrimenti verrà sostituita solo la prima.

Example 5.2 Sostituire del testo nella riga corrente

Il tuo caro vecchio compilatore ha appena finito di bloccarsi dicendoti che la classe `myClass`, che compare nella riga 3902 del tuo file sorgente, non è definita.

"Naturale!" pensi, sai benissimo che si scrive `MyClass`. Vai alla riga 3902 e, invece di provare a cercare la parola nel testo, apri la finestra dei comandi di modifica, inserisci **s/myclass/MyClass/i**, premi **OK**, salvi il file e lo compili – con successo senza errori.

Example 5.3 Sostituire del testo nell'intero file

Immagina di avere un file, nel quale nomi 'Miss Jensen' parecchie volte, e che arriva qualcuno che ti dice che si è appena sposata con 'Mr Jones'. Naturalmente tu vuoi sostituire tutti i 'Miss Jensen' con 'Ms Jones'.

Vai alla riga di comando e digita il comando **%s/Miss Jensen/Ms Jones/** e premi invio. Il gioco è fatto!

Example 5.4 Un esempio più avanzato

Questo esempio usa i *referimenti all'indietro* e le *classi di carattere* (se non sai che cosa sono, leggi la relativa documentazione citata più in basso).

Immagina di avere la seguente riga:

```
void MyClass::DoStringOps( String      &foo, String &bar, String *p, int  & ←
    a, int &b )
```

Ora ti rendi conto che il codice non va bene, e decidi di usare la parola chiave `const` per tutti gli argomenti 'reference', quelli caratterizzati dall'operatore `&` posto davanti al nome dell'argomento. Inoltre vorresti semplificare lo spazio vuoto, in modo che ci sia solo uno spazio tra ogni parola.

Apri la finestra dei comandi di modifica e inserisci: **s/\s+(\w+)\s+(\&)/ const \1 \2/g** e premi il pulsante **OK**. La **g** che alla fine dell'espressione fa ricompilare l'espressione regolare per ogni corrispondenza per salvarne i *referimenti all'indietro*.

Otteni: `void MyClass::DoStringOps(const String &foo, const String &bar, String *p, const int &a, const int &b)`

Missione compiuta! Ora cos'è successo? Abbiamo cercato alcuni spazi vuoti (`\s+`) seguiti da alcuni caratteri alfabetici (`\w+`), seguiti da altri spazi vuoti (`\s+`), seguiti da una `e` commerciale e la porzione alfabetica con la `e` commerciale salvati nel processo per essere usati nell'operazione di sostituzione. Quindi abbiamo sostituito la parte corrispondente della nostra riga con uno spazio vuoto seguito da 'const', seguito da un altro spazio vuoto, seguito dalla porzione alfabetica (`\1`), seguito da uno spazio vuoto e infine seguito dalla `e` commerciale salvata (`\2`)

Ora, in alcuni casi la porzione alfabetica era una 'String'. in altri 'int', quindi se usi `\w` e il quantificatore `+` sarà meglio.

sort

Ordina il testo selezionato o tutto il documento.

natsort

Ordina le righe selezionate o tutto il documento naturalmente.

Example 5.5 sort e natsort

sort(a10, a1, a2) produce a1, a10, a2.

natsort(a10, a1, a2) produce a1, a2, a10.

moveLinesDown

Sposta le righe selezionate in basso.

moveLinesUp

Sposta le righe selezionate in alto.

uniq

Rimuovi le righe duplicate dal testo selezionato o tutto il documento.

rtrim

Rimuovi gli spazi finali dal testo selezionato o tutto il documento.

ltrim

Rimuovi gli spazi iniziali dal testo selezionato o tutto il documento.

join [STRINGA separatore]

Unisce le righe selezionate o tutto il documento. A scelta, prende un parametro che definisce un separatore, per esempio **join ' , '**.

rmblank

Rimuove tutti gli spazi vuoti dal testo selezionato o da tutto il documento.

alignon

Questo comando allinea, alla colonna data da un'espressione regolare data come argomento, le righe nel blocco selezionato o nell'intero documento.

Per impostazione predefinita, se inserisci un modello vuoto l'allineamento sarà col primo carattere non di spazio.

Se il modello ha una cattura, il rientro sarà con la prima corrispondenza catturata.

Esempi:

alignon - inserirà degli spazi prima del primo «-» di ogni riga per allinearle nella stessa colonna.

alignon :\\s+(.) inserirà degli spazi prima del primo carattere non di spazio che ricorre dopo i due punti per allinearle nella stessa colonna.

unwrap

Unisce il testo selezionato o tutto il documento.

each STRINGA script

Data una funzione JavaScript come argomento, la chiama per l'elenco di righe selezionate e le sostituisce con il valore restituito dalla funzione.

Example 5.6 Unire le righe selezionate

each 'function(lines){return lines.join(", ")}'

O più brevemente:

each 'lines.join(", ").'

filter STRINGA script

Data una funzione JavaScript come argomento, la chiama per l'elenco di righe selezionate e rimuove quelle per cui la funzione restituisce falso.

Example 5.7 Rimuovere le righe vuote

```
filter 'function(1){return 1.length > 0;}'
```

O più brevemente:

```
filter 'line.length > 0'
```

map STRINGA script

Data una funzione JavaScript come argomento, la chiama per l'elenco di righe selezionate e sostituisce la riga con il valore della funzione.

Example 5.8 Rimuovere le righe vuote

```
map 'function(line){return line.replace(/^s+/, "");}'
```

O più brevemente:

```
map 'line.replace(/^s+/, "")'
```

duplicateLinesUp

Duplica le righe selezionate sopra la selezione attuale.

duplicateLinesDown

Duplica le righe selezionate sotto la selezione attuale.

5.2.1.3 Comandi per gli spostamenti

goto INT riga

Questo comando sposta il cursore alla riga indicata.

grep STRINGA modello

Cerca nel documento l'espressione regolare **modello**. Per maggiori informazioni, vedi appendice [A](#).

find STRINGA modello

Questo comando si sposta alla prima corrispondenza del **modello** secondo la configurazione. Le corrispondenze successive possono essere trovate usando **Modifica** → **Trova successivo** (la scorciatoia predefinita è F3).

Il comando **find** può essere configurato aggiungendo in fondo un due punti seguito da una o più opzioni, nella forma **find:opzioni modello**. Le seguenti opzioni sono disponibili:

- b** Cerca all'indietro.
- c** Cerca dalla posizione del cursore.
- e** Cerca solo all'interno della selezione.
- r** Esegue la ricerca con un'espressione regolare. Se è impostata, puoi usare **\N**, dove N è un numero che rappresenta le catture nella stringa di sostituzione.
- s** Esegue la ricerca facendo distinzione fra maiuscole e minuscole.

w Fa in modo che la ricerca consideri solo parole intere.

ifind STRINGA modello

Questo comando fornisce la ricerca 'mentre scrivi'. Puoi configurarne il comportamento aggiungendo in fondo un due punti seguito da una o più opzioni, nella forma **ifind:opzioni modello**. Le seguenti opzioni sono disponibili:

b Cerca all'indietro.
r Esegue la ricerca di un'espressione regolare.
s Esegue la ricerca facendo distinzione fra maiuscole e minuscole.
c Cerca dalla posizione del cursore.

5.2.1.4 Comandi per le funzioni fondamentali dell'editor (dipendono dall'applicazione in cui il componente editor è in uso)

w Salva il documento attuale.
wa Salva tutti i documenti attualmente aperti.
q Chiudi il documento attuale.
qa Chiudi tutti i documenti aperti.
wq Salva e chiudi il documento attuale.
wqa Salva e chiudi tutti i documenti attualmente aperti.
x Salva e chiudi il documento attuale solo se è stato modificato.
x Salva e chiudi tutti i documenti attualmente aperti solo se sono stati modificati.
bp Vai al documento precedente nell'elenco dei documenti.
bn Vai al documento successivo nell'elenco dei documenti.
new Apri un nuovo documento nella vista separata orizzontalmente.
vnew Apri un nuovo documento nella vista separata verticalmente.
e Ricarica il documento attuale se è stato modificato sul disco.
enew Modifica un nuovo documento.
print Apri la finestra di stampa per stampare il documento attuale.

5.3 Uso del raggruppamento del codice

Il raggruppamento del codice permette di nascondere parti del documento nell'editor, rendendo più facile una panoramica di documenti di grandi dimensioni. In KatePart le regioni raggruppabili sono classificate usando le regole delle definizioni dell'evidenziazione della sintassi, e quindi sono disponibili solo in alcuni formati — tipicamente codice sorgente, annotazioni XML e simili. La maggior parte delle definizioni di evidenziazione che permettono il raggruppamento del codice permettono anche la definizione manuale di regioni raggruppabili, tipicamente con l'uso delle parole chiave **BEGIN** ed **END**.

Per usare la funzione di raggruppamento del codice, attiva gli indicatori di raggruppamento usando la voce **Visualizza** → **Mostra segni di raggruppamento** del menu se non sono già visibili. Il pannello degli indicatori di raggruppamento sul lato sinistro dello schermo mostra una vista grafica delle regioni raggruppabili, con simbolo di un triangolo indicante l'operazione possibile su una data regione: Un triangolo capovolto indica che la regione è espansa, facendo clic su di esso si ridurrà la regione ad una sola riga e successivamente verrà mostrato come simbolo un triangolo diritto.

Sono forniti tre comandi per manipolare lo stato delle regioni di raggruppamento del codice, vedi la [documentazione del menu](#).

Le righe raggruppate vengono ricordate alla chiusura del file, così quando lo riapri i nodi raggruppati lo saranno ancora. Questo vale anche per le operazioni di ricarica.

Se non vuoi usare la funzione di raggruppamento del codice, puoi disabilitare l'opzione **Mostra segni di raggruppamento (se disponibile)** dalla pagina [Aspetto](#) dalla pagina di configurazione dell'editor.

Capitolo 6

Estensione di KatePart

T.C. Hollingsworth

Traduzione della documentazione.: Nicola Ruggero

Aggiornamento per Kate 2.5.6: Luciano Montanaro

Aggiornamento e manutenzione della traduzione: Paolo Zamponi

6.1 Introduzione

Come ogni componente editor di testi avanzato, KatePart offre una moltitudine di modi per espandere le proprie funzionalità: puoi [scrivere semplici script per aggiungere funzionalità con JavaScript](#). Infine, dopo aver esteso KatePart saresti il benvenuto [nel nostro gruppo](#), per condividere il tuo lavoro con il mondo intero!

6.2 Lavorare con l'evidenziazione della sintassi

6.2.1 Panoramica

L'evidenziazione della sintassi è quel qualcosa che rende l'editor in grado di visualizzare automaticamente il testo con stili e colori differenti, a seconda delle funzioni delle stringhe in relazione allo scopo del file. Nel sorgente di un programma, per esempio, le istruzioni di controllo potrebbero essere rese in grassetto, mentre i tipi di dati e i commenti in colori diversi dal resto del testo. Questo migliora di molto la leggibilità del testo, e rende l'autore più efficiente e più produttivo.

```
Theme Repository::defaultTheme(Repository::DefaultTheme t)
{
    if (t == DarkTheme)
        return theme(QLatin1String("Breeze Dark"));
    return theme(QLatin1String("Default"));
}
```

Una funzione C++, resa con l'evidenziazione della sintassi.

```
Theme Repository::defaultTheme(Repository::DefaultTheme t)
{
    if (t == DarkTheme)
        return theme(QLatin1String("Breeze Dark"));
    return theme(QLatin1String("Default"));
}
```

La stessa funzione C++, senza l'evidenziazione.

Dei due esempi, qual è il più semplice da leggere?

KatePart nasce con un sistema flessibile e configurabile che è in grado di eseguire l'evidenziazione della sintassi. Già nella distribuzione standard vengono fornite alcune definizioni che coprono una vasta gamma di linguaggi di programmazione, di scripting e di markup e per altri formati di file di testo; puoi inoltre inserire le tue definizioni in un semplice file XML.

Quando apri un file basato sul tipo MIME del file KatePart, riconoscerà automaticamente le giuste regole della sintassi; questo in base alla sua estensione o, se non presente, al suo contenuto. Puoi sempre impostare manualmente la sintassi da usare, qualora notassi dei problemi, facendo clic su **Strumenti** → **Evidenziazione**.

Gli stili e i colori usati da ciascuna definizione di evidenziazione della sintassi possono essere configurati usando la scheda [Stili di testo evidenziato](#), mentre i tipi MIME e le estensioni del file che dovrebbero essere usate a questo scopo sono gestiti nella scheda [Modi e tipi di file](#).

NOTA

L'evidenziazione della sintassi serve per migliorare la leggibilità del testo, ma non puoi affidarti ad essa per convalidarlo. La marcatura del testo in base alla sintassi può essere difficoltosa a seconda del formato che stai utilizzando: in qualche caso l'autore delle regole della sintassi sarà orgoglioso se il 98% del testo venisse renderizzato correttamente. Ad ogni modo, in genere, sarà difficile incontrare quel 2% non corretto.

6.2.2 Il sistema di evidenziazione della sintassi di KatePart

In questa sezione sarà trattato più in dettaglio il meccanismo di evidenziazione della sintassi. Ti servirà se vorrai saperne di più, oppure se vorrai cambiare o creare delle definizioni di sintassi.

6.2.2.1 Come funziona

Una delle prime cose che l'editor KatePart fa ogni volta che apri un file è quella di individuare la definizione di sintassi da usare. Mentre stai leggendo il testo del file, e mentre stai digitando qualcosa, il sistema di evidenziazione della sintassi analizza il testo usando le regole definite dalle definizioni di sintassi, e lo marca dove i diversi contesti e gli stili iniziano e finiscono.

Quando digiti nel documento, il nuovo testo viene analizzato e marcato al volo. In questo modo, se cancelli un carattere che è marcato come inizio o fine di un contesto, lo stile del testo circostante viene modificato di conseguenza.

Le definizioni di sintassi usate dal sistema di evidenziazione di KatePart sono dei file XML, contenenti

- Regole per individuare il ruolo del testo, organizzate in blocchi di contesto
- Liste di parole chiave
- Definizioni di oggetti di stile

Quando il testo viene analizzato, le regole di rilevamento vengono valutate nell'ordine in cui sono definite: se l'inizio della stringa corrente coincide con una qualche regola, allora viene usato il relativo contesto. Il punto di inizio del testo viene spostato nel punto finale in cui la regola è stata applicata; inizia così un nuovo ciclo, che comincia nel contesto impostato dalla regola applicata.

6.2.2.2 Regole

Le regole di rilevamento sono il cuore del sistema di rilevamento della sintassi. Una regola è una stringa, un carattere o un'espressione regolare con cui confrontare il testo che viene analizzato. Contiene le informazioni sullo stile da usare per la parte corrispondente del testo. Può passare il contesto di lavoro del sistema sia ad un contesto menzionato esplicitamente che al contesto precedente usato dal testo.

Le regole sono organizzate in gruppi di contesto, ciascuno dei quali viene usato per i concetti principali di testo all'interno del formato, per esempio le stringhe di testo virgolettato o i blocchi di commenti nei sorgenti dei programmi. Questo assicura che il sistema di evidenziazione non abbia la necessità di cercare tra le regole se non è necessario, e che alcune sequenze di caratteri nel testo possono essere trattate in maniera diversa a seconda del contesto corrente.

I contesti possono essere generati dinamicamente per permettere l'uso di istanze di dati specifici nelle regole.

6.2.2.3 Stili di contesto e parole chiave

In alcuni linguaggi di programmazione i numeri interi sono trattati dal compilatore (il programma che converte il codice sorgente in binario eseguibile) in maniera diversa da quelli in virgola mobile, inoltre all'interno di una stringa virgolettata ci possono essere dei caratteri con un significato speciale. In questi casi ha senso visualizzarli in modo differente da quelli circostanti, in modo che possano essere identificati facilmente durante la lettura del testo. Così, pur non rappresentando un contesto speciale, questi possono essere comunque trattati come tali dal sistema di evidenziazione della sintassi, in modo da essere contrassegnati e quindi visualizzati in modo diverso.

Una definizione di sintassi può contenere tutti gli stili richiesti per racchiudere i concetti del formato per cui è usata.

In molti formati ci sono liste di parole che rappresentano un concetto specifico: per esempio nei linguaggi di programmazione le istruzioni di controllo sono un concetto, i nomi dei tipi di dato un altro, e le funzioni proprie del linguaggio un terzo. Il sistema di evidenziazione della sintassi di KatePart può usare queste liste per individuare le parole marcandole nel testo, in modo da enfatizzare i concetti dei formati di testo.

6.2.2.4 Stili predefiniti

Se apri in KatePart un file sorgente in C++, un sorgente Java™ o un documento HTML vedrai che, anche se i formati e le parole scelte per il trattamento speciale sono diversi, i colori utilizzati sono gli stessi. Questo perché KatePart ha una lista di stili predefiniti che vengono impiegati dalle definizioni individuali di sintassi.

Questo rende semplice riconoscere concetti simili in formati di testo diversi. Per esempio, i commenti sono presenti in quasi tutti i linguaggi di programmazione, di scripting e di markup: se sono sempre renderizzati utilizzando lo stesso stile in tutti i linguaggi, allora non avrai bisogno di fermarti a pensare per identificarli nel testo.

SUGGERIMENTO

Tutti gli stili di una definizione di sintassi usano uno degli stili predefiniti: solo poche di esse fanno uso di altri stili non predefiniti. Potrebbe essere il caso di lanciare la finestra di configurazione se usi spesso un formato, così da vedere se alcuni concetti fanno uso di uno stesso stile. Per esempio, c'è solo uno stile predefinito per le stringhe, ma il linguaggio di programmazione Perl ha due tipi di stringhe: potresti quindi migliorarne l'evidenziazione, configurandoli in modo che siano leggermente diversi. Tutti gli [stili predefiniti disponibili](#) verranno spiegati in seguito.

6.2.3 Il formato XML di definizione dell'evidenziazione

6.2.3.1 Panoramica

KatePart usa l'infrastruttura di evidenziazione della sintassi di KDE Frameworks. I file XML per l'evidenziazione predefinita forniti insieme a KatePart sono compilati in modo predefinito nella libreria di evidenziazione della sintassi.

Questa sezione è una panoramica sul formato XML di definizione dell'evidenziazione. Basandosi su un piccolo esempio saranno descritti i componenti principali, il loro significato e il loro uso. La prossima sezione scenderà invece nei dettagli delle regole di rilevamento dell'evidenziazione.

La definizione formale, anche conosciuta come XSD, si trova nel [deposito dell'evidenziazione della sintassi](#), nel file `language.xsd`

I file `.xml` per la definizione personalizzata di evidenziazione si trovano in `org.kde.syntax-highlighting/syntax/` nella tua cartella utente, e vengono trovati con **`qtpaths--paths GenericDataLocation`**, che generalmente sono `$HOME /.local/share/` e `/usr/share/`.

Per i pacchetti Flatpak e Snap la cartella qui sopra non funzionerà perché la posizione dei dati è diversa per ogni applicazione. In un'applicazione Flatpak la posizione dei file XML personalizzati è generalmente `$HOME /.var/app/ nome-del-pacchetto-flatpak /data/org.kde.syntax-highlighting/syntax/`, mentre in una Snap la posizione è `$HOME /snap/ snap-package-name /current/.local/share/org.kde.syntax-highlighting/syntax/`.

In Windows® questi file si trovano in `%USERPROFILE%\AppData\Local\org.kde.syntax-highlighting\syntax`. `%USERPROFILE%` viene generalmente espanso in `C:\Users\utente`.

In sintesi, per la maggior parte delle configurazioni la cartella dei file XML personalizzati è la seguente:

Per l'utente locale	<code>\$HOME /.local/share/org.kde.syntax-highlighting/syntax/</code>
Per tutti gli utenti	<code>/usr/share/org.kde.syntax-highlighting/syntax/</code>
Per i pacchetti Flatpak	<code>\$HOME /.var/app/ nome-pacchetto-flatpak /data/org.kde.syntax-highlighting/syntax/</code>
Per i pacchetti Snap	<code>\$HOME /snap/ nome-pacchetto-snap /current/.local/share/org.kde.syntax-highlighting/syntax/</code>
In Windows®	<code>%USERPROFILE%\AppData\Local\org.kde.syntax-highlighting\syntax</code>
In macOS®	<code>\$HOME /Library/Application Support/org.kde.syntax-highlighting/syntax/</code>

Se esistono più file per lo stesso linguaggio verrà caricato quello che ha il valore più alto dell'attributo **version** nell'elemento **language**.

LE SEZIONI PRINCIPALI DEI FILE DI DEFINIZIONE DELL'EVIDENZIAMENTO DI KATEPART

Un file di evidenziazione contiene un'intestazione che imposta la versione XML:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Il file di definizione inizia con l'elemento **language**. Gli attributi disponibili sono:

Attributi richiesti:

name imposta il nome del linguaggio. Appare successivamente nei menu e nelle finestre.

section specifica la categoria.

extensions definisce le estensioni dei file, come `"*.cpp;*.h"`

version specifica l'attuale revisione del file di definizione in termini di un numero intero; sii certo di incrementare questo numero se cambi un file di definizione dell'evidenziazione.

kateversion specifica l'ultima versione supportata di KatePart.

Attributi opzionali:

mimetype associa i tipi MIME dei file.

casesensitive definisce se le parole chiave sono sensibili alle maiuscole oppure no.

priority è necessaria se un'altra definizione di evidenziazione usa le stesse estensioni; vince quella con priorità più alta.

author contiene il nome dell'autore e il suo indirizzo di posta elettronica.

license contiene la licenza, di solito MIT, per i nuovi file di evidenziazione della sintassi.

style contiene il linguaggio fornito, ed è usato dai rientratori per gli attributi `required-syntax-style`.

rientratore definisce quale rientratore sarà usato come predefinito. Quelli disponibili sono: *ada*, *normal*, *cstyle*, *cmake*, *haskell*, *latex*, *lilypond*, *lisp*, *lua*, *pascal*, *python*, *replicode*, *ruby* e *xml*.

hidden definisce se il nome deve apparire nei menu di KatePart.

Allora la prossima riga potrebbe apparire così:

```
<language name="C++" version="1" kateversion="2.4" section="Sources" ←  
  extensions="*.cpp;*.h" />
```

Viene poi l'elemento **highlighting**, che contiene l'elemento opzionale **list**, e i richiesti **contexts** e **itemDatas**.

L'elemento **list** contiene una lista di parole chiave; in questo caso le parole chiave sono *class* e *const*; puoi aggiungere tutte le liste di cui hai bisogno.

Da KDE Frameworks 5.53 una lista può includere delle parole chiave di un'altra lista oppure da un'altra lingua (o da un altro file) usando l'elemento **include**. Viene usato **##** per separare il nome della lista da quello del linguaggio di definizione del nome, come per il ruolo **IncludeRules**. Ciò è utile per evitare la duplicazione di liste delle parole se hai bisogno di includere le parole chiave di un altro linguaggio o file. Ad esempio, la lista *other-name* contiene la parola chiave *str* e tutte le parole chiave della lista *types* che appartiene al linguaggio *ISO C++*.

L'elemento **contexts** contiene tutti i contesti; il primo è di default l'inizio dell'evidenziazione. Ci sono due regole nel contesto *Normal Text*, una associa la lista delle parole chiave con nome *somename*, e un'altra rileva una virgoletta, passando il contesto a *string*. Per saperne di più leggi il prossimo capitolo.

La terza parte è l'elemento **itemDatas**: contiene tutti i colori e gli stili dei caratteri che sono necessari per il contesto e per le regole. In questo esempio vengono usati **itemData** *Normal Text*, *String* e *Keyword*.

```
<highlighting>
  <list name="somename">
    <item>class</item>
    <item>const</item>
  </list>
  <list name="othername">
    <item>str</item>
    <include>types##ISO C++</include>
  </list>
  <contexts>
    <context attribute="Normal Text" lineEndContext="#pop" name=" ←
      Normal Text" >
      <keyword attribute="Keyword" context="#stay" String="somename" ←
        />
      <keyword attribute="Keyword" context="#stay" String="othername" ←
        />
      <DetectChar attribute="String" context="string" char="&quot;" ←
        />
    </context>
    <context attribute="String" lineEndContext="#stay" name="string" ←
      >
      <DetectChar attribute="String" context="#pop" char="&quot;" />
    </context>
  </contexts>
  <itemDatas>
    <itemData name="Normal Text" defStyleNum="dsNormal" />
    <itemData name="Keyword" defStyleNum="dsKeyword" />
    <itemData name="String" defStyleNum="dsString" />
  </itemDatas>
</highlighting>
```

L'ultima parte di una definizione di evidenziazione è la sezione opzionale **general**. Può contenere informazioni sulle parole chiave, sul raggruppamento del codice, sui commenti, sui rientri, sulle righe vuote e sul controllo ortografico.

La sezione **comment** definisce con quale stringa viene introdotta una singola riga di commento. Puoi anche definire un commento multi-riga usando *multiLine* con l'attributo addizionale *end*. Questo viene usato se l'utente preme la scorciatoia per *commentare/non commentare*.

La sezione **keywords** definisce se la lista delle parole chiave è sensibile alle maiuscole oppure no. Altri attributi verranno spiegati più tardi.

Le altre sezioni, **folding**, **emptyLines** e **spellchecking** sono di solito non necessarie e verranno spiegate in seguito.

```
<general>
  <comments>
    <comment name="singleLine" start="##"/>
    <comment name="multiLine" start="####" end="####" region=" ←
      CommentFolding"/>
  </comments>
  <keywords casesensitive="1"/>
  <folding indentationsensitive="0"/>
  <emptyLines>
    <emptyLine regexpr="\s+"/>
    <emptyLine regexpr="\s*#.*/>
  </emptyLines>
  <spellchecking>
    <encoding char="á" string="\`a"/>
```

```
<encoding char="à" string="\`a"/>
</spellchecking>
</general>
</language>
```

6.2.3.2 Le sezioni in dettaglio

Questa parte descriverà tutti gli attributi disponibili per contexts, itemDatas, keywords, comments, il raggruppamento del codice e i rientri.

L'elemento **context** appartiene al gruppo **contexts**. Di per sé un contesto definisce le regole specifiche del contesto, tipo cosa dovrebbe accadere qualora il sistema di evidenziazione raggiunga il fine riga. Gli attributi disponibili sono:

name dichiara il nome del contesto. Le regole faranno uso di questo nome per specificare il contesto al quale passare se la regola viene soddisfatta.

attribute identifica lo stile da usare per un carattere quando nessuna regola corrisponde o quando queste non specificano attributi. Nell'ultimo caso verrà usato l'**attribute** del contesto specificato nel *context* della regola.

lineEndContext definisce il contesto a cui il sistema di evidenziazione passa se viene raggiunta la fine della riga. Questo può essere sia il nome di un altro contesto, **#stay** per non lasciare il contesto (ad es.. non fare niente), oppure **#pop**, che causerà l'abbandono di questo contesto. È possibile usare per esempio **#pop#pop#pop** per saltare tre volte, o anche **#pop#pop!OtherContext** per saltare due volte e passare al contesto chiamato **OtherContext**. È possibile passare anche ad un contesto che appartiene ad un'altra definizione di linguaggio, come in **IncludeRules**, ad es. **SomeContext##JavaScript** non è valido. I cambi di contesto sono descritti anche in Sezione 6.2.4. Predefinito: **#stay**.

lineEmptyContext definisce il contesto se viene incontrata una riga vuota. Predefinito: **#stay**. La nomenclatura dei cambi di contesto è la stessa che è stata precedentemente descritta in *lineEndContext*. Predefinito: **#stay**.

fallthroughContext specifica il contesto successivo a cui passare se nessuna regola viene soddisfatta. La nomenclatura dei cambi di contesto è la stessa che era stata descritta in precedenza in *lineEndContext*. Predefinito: **#stay**.

fallthrough definisce se il sistema di evidenziazione passa al contesto specificato in **fallthroughContext** se non viene soddisfatta nessuna regola. Note che da KDE Frameworks 5.62 questo attributo è deprecato in favore di **fallthroughContext**, dal momento che se l'attributo **fallthroughContext** è presente, è implicitamente inteso che il valore di **fallthrough** è **true**. Predefinito: **false**.

noIndentationBasedFolding disabilita il raggruppamento basato sui rientri nel contesto. Se il raggruppamento basato sui rientri non è attivato questo attributo è inutile. Esso è definito nell'elemento *folding* del gruppo *general*. Predefinito: **false**.

L'elemento **itemData** appartiene al gruppo **itemDatas**. Definisce lo stile del carattere e i colori, e quindi è possibile definire i propri stili e i propri colori. Raccomandiamo però di attenersi agli stili predefiniti, se possibile, cosicché gli utenti vedranno sempre gli stessi colori anche in linguaggi diversi. Tuttavia a volte non c'è altro modo se non quello di cambiare i colori e gli attributi dei caratteri. Sono richiesti il nome degli attributi e **StyleNumDef**, ma gli altri sono opzionali. Gli attributi disponibili sono:

name imposta il nome di itemData. I contesti e le regole useranno questo nome nei loro attributi *attribute* a cui fa riferimento itemData.

defStyleNum definisce quale stile predefinito usare. Gli stili predefiniti disponibili sono spiegati in dettaglio più tardi.

color definisce un colore. Sono validi i formati «#rrggbb» o «#rgb».

selColor definisce la selezione del colore.
Il testo sarà in corsivo se **italic** è *true*.
Il testo sarà in grassetto se **bold** è *true*.
Il testo sarà sottolineato se **underline** è *true*.
Il testo sarà in barrato se **strikeOut** è *true*.
Il testo sarà controllato ortograficamente se **spellChecking** è *true*.

L'elemento **keywords** del gruppo **general** definisce le proprietà delle parole chiave. Gli attributi disponibili sono:

casesensitive può essere *true* oppure *false*; se è *true* tutte le parole chiave soddisfano la sensibilità alle maiuscole.
weakDelimiter è una lista di caratteri che non agiscono da delimitatori di parole; per esempio, il punto «.» è un delimitatore di parola. Se assumiamo che una parola chiave in una **lista** possa contenere un punto, sarà soddisfatta solo se specifichi che il punto è un delimitatore debole.
additionalDelimiter definisce delimitatori addizionali.
wordWrapDelimiter definisce i caratteri dopo dei quali una riga può andare a capo.
Sono delimitatori predefiniti e terminatori di parola i caratteri `. () : ! + , - < = > % & * / ; ? [] ^ { | } ~ \`, gli spazi (`<< >>`) e i tabulatori (`<<\t>>`).

L'elemento **comment** del gruppo **comments** definisce le proprietà che sono usate in Strumenti → Commenta, Strumenti → Decomenta e Strumenti → Commenta/decommenta. Gli attributi disponibili sono:

name può essere *singleLine* o *multiLine*. Se scegli *multiLine* sono richiesti gli attributi *end* e *region*. Se scegli *singleLine* puoi aggiungere l'attributo facoltativo *position*.
start definisce la stringa usata per iniziare un commento. In C++ sarebbe `"/**` nei commenti multi-riga. Questo attributo è richiesto per i tipi *multiLine* e *singleLine*.
end definisce la stringa per chiudere un commento. In C++ questo sarebbe `*/`. Questo attributo è solo disponibile ed è richiesto per i commenti di tipo *multiLine*.
region dovrebbe essere il nome dei commenti multi-riga raggruppabili. Assumendo di avere nelle regole `beginRegion="Comment" ... endRegion="Comment"` si dovrebbe usare `region="Comment"`. In questo modo l'azione di decommento funziona anche se non viene selezionato tutto il testo di un commento multi-riga, il cursore deve solo essere all'interno del commento. Questo attributo è disponibile solo per il tipo *multiLine*.
position definisce dove vengono inseriti i commenti di una sola riga. Per impostazione predefinita essi sono posizionati all'inizio della riga, alla colonna 0, ma se utilizzi `position="afterwhitespace"` il commento viene inserito dopo i primi spazi sulla destra, prima del primo carattere non di spaziatura. Ciò è utile per inserire correttamente i commenti nei linguaggi in cui il rientro è importante, ad esempio Python o YAML. Questo attributo è facoltativo, e l'unico valore possibile è *afterwhitespace*. È disponibile solo per il tipo *singleLine*.

L'elemento **raggruppamento** del gruppo **general** definisce le proprietà di raggruppamento del codice. Gli attributi disponibili sono:

Se **indentationsensitive** è *true* i marcatori del raggruppamento del codice saranno aggiunti basandosi sul rientro, come nel linguaggio di scripting Python. Di solito non hai necessità di impostarlo, e di solito è impostato a *false*.

L'elemento **emptyLine** nel gruppo **emptyLines** definisce quali righe dovrebbero essere trattate come vuote. Ciò permette di modificare il comportamento dell'attributo *lineEmptyContext* negli elementi **context**. Gli attributi disponibili sono:

regexpr definisce un'espressione regolare che verrà trattata come una riga vuota. Per impostazione predefinita le righe vuote non contengono nessun carattere, perciò esso aggiunge delle altre righe vuote, ad esempio se vuoi considerare come vuote anche le righe con

gli spazi. Ad ogni modo, nella maggior parte delle definizioni di sintassi non è necessario impostare questo attributo.

L'elemento **encoding** del gruppo **spellchecking** definisce una codifica dei caratteri per il controllo ortografico. Gli attributi disponibili sono:

char è un carattere codificato.

string è una sequenza di caratteri che verrà codificata come il carattere *char* nel controllo ortografico. Ad esempio, nel linguaggio LaTeX la stringa `\~{A}` rappresenta il carattere **Ä**.

6.2.3.3 Stili predefiniti disponibili

Gli stili predefiniti sono stati [già trattati](#), come piccolo riassunto: gli stili predefiniti sono i caratteri predefiniti e gli stili colore.

Stili generali predefiniti:

dsNormal, quando non è richiesta nessuna evidenziazione speciale.

dsKeyword, parola chiave integrata nel linguaggio.

dsFunction, chiamate di funzioni e definizioni.

dsVariabile, se applicabile: nomi delle variabili (ad es. `$someVar` in PHP/Perl).

dsControlFlow, parole chiave per il controllo del flusso, come `if`, `else`, `switch`, `break`, `return`, `yield`, ...

dsOperator, operatori come `+`, `-`, `*`, `/`, `::`, `<`, `>`

dsBuiltIn, funzioni integrate, classi e oggetti.

dsExtension, estensioni comuni, come le classi `Qt™` e le funzioni o le macro in C++ e in Python.

dsPreprocessor, direttive al preprocessore o definizioni di macro.

dsAttribute, annotazioni come `@override` e `__declspec(...)`.

Stili predefiniti relativi alle stringhe:

dsChar, caratteri singoli, come `'x'`.

dsSpecialChar, caratteri con significati speciali nelle stringhe, come `escape`, sostituzioni, oppure operatori regex.

dsString, stringhe come `"hello world"`.

dsVerbatimString, letterale o stringa grezza, come `'raw \backslash'` in Perl, CoffeeScript, e shell, come `r'\raw'` in Python.

dsSpecialString, SQL, regexes, documenti HERE, modi $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ `math`, ...

dsImport, importazione, inclusione, richieste di moduli.

Stili predefiniti relativi ai numeri:

dsDataType, tipi di dati integrati, come `int`, `void`, `u64`.

dsDecVal, valori decimali.

dsBaseN, valori con base diversa da 10.

dsFloat, valori in virgola mobile.

dsConstant, costanti integrate e definite dall'utente, come `PI`.

Commenti e stili predefiniti relativi alla documentazione:

dsComment, commenti.
dsDocumentation, `/**` Commenti nella documentazione `*/` o `"""` docstrings `"""`.
dsAnnotation, comandi nella documentazione, come `@param`, `@brief`.
dsCommentVar, i nomi delle variabili usati nei comandi precedenti, come "pippobar" in `@param pippobar`.
dsRegionMarker, delimitatori di regioni, come `//BEGIN` e `//END` nei commenti.

Altri stili predefiniti:

dsInformation, annotazioni e suggerimenti, come `@note` in doxygen.
dsWarning, avvisi, come `@warning` in doxygen.
dsAlert, parole speciali, come `TODO`, `FIXME`, `XXXX`.
dsError, evidenziazione di errori e sintassi errata.
dsOthers per tutto il resto.

6.2.4 Regole di rilevamento dell'evidenziazione

Questa sezione descrive le regole di rilevamento della sintassi.

Ogni regola può associare zero o più caratteri all'inizio della stringa su cui è testata. Se la regola corrisponde i caratteri associati sono assegnati allo stile o all'*attributo* definito dalla regola; una regola può chiedere che l'attuale contesto venga cambiato.

Una regola è un qualcosa che assomiglia a questo:

```
<RuleName attribute="(identifier)" context="(identifier)" [attributi ←  
specifici della regola] />
```

attribute identifica lo stile da usare per i caratteri associati secondo il nome, mentre *context* identifica il contesto da usare da qui in avanti.

context può essere identificato da:

- *identifier*, che è il nome dell'altro contesto.
- *order*, che dice al motore di rimanere nel contesto attuale (**#stay**), oppure di ritornare a quello precedentemente usato nella stringa (**#pop**). Un contesto vuoto o assente è equivalente a **#stay**.
Per tornare indietro di più passi la parola chiave **#pop** può essere ripetuta: **#pop#pop#pop**
- *order* seguito da un punto esclamativo (!) e *identifier*, che faranno prima seguire al motore l'ordine, e poi lo mandano in un altro contesto, ad es. **#pop#pop!OtherContext**.
- Un identificatore, che è un nome di contesto, seguito da due cancelletti (##) e da un altro *identificatore*, cioè dal nome di una definizione di linguaggio. Questa denominazione è simile a quella usata nelle regole **IncludeRules**, e ti permette di passare ad un contesto appartenente ad un'altra definizione di evidenziazione della sintassi, ad esempio **SomeContext##JavaScript**.

Gli attributi specifici per la regola variano, e sono descritti nelle sezioni seguenti.

ATTRIBUTI COMUNI

- *attribute*: un attributo associato ad un determinato *itemData*. Predefinito: *attribute* dal contesto specificato nell'attributo *context*.
- *context*: specifica il contesto al quale il sistema di evidenziazione passa se la regola corrisponde. Predefinito: **#stay**.

- *beginRegion*: inizia un blocco di raggruppamento del codice. Predefinito: non impostato.
- *endRegion*: chiude un un blocco di raggruppamento del codice. Predefinito: non impostato.
- *lookAhead*: se impostato a *true* il sistema di evidenziazione non processerà la lunghezza assegnata. Predefinito: *false*.
- *firstNonSpace*: corrisponde solo se la stringa è la prima senza spazi nella riga. Predefinito: *false*.
- *column*: corrisponde solo se la colonna combacia. Predefinito: non impostato.

REGOLE DINAMICHE

- *dynamic*: può essere (*true* | *false*).

Come funziona:

Nelle [espressioni regolari](#) delle regole **RegExpr**, tutto il testo all'interno delle parentesi curve semplici (**MODELLO**) viene catturato e ricordato. Queste catture possono essere usate nel contesto nel quale ci si è spostati, nelle regole con l'attributo **dynamic vero**, da *%N* (in *String*) oppure *N* (in *char*).

È importante ricordare che il testo catturato in una regola **RegExpr** viene memorizzato solo per il contesto nel quale si è passati, specificato nel suo attributo **context**.

SUGGERIMENTO

- Se le catture non vengono usate, nelle regole dinamiche o nell'espressione regolare stessa, si dovrebbe usare **gruppi non di cattura**: **(?=MODELLO)**
I gruppi *lookahead* o *lookbehind* come **(?=MODELLO)** o **(?!MODELLO)** non vengono catturati. Vedi [espressioni regolari](#) per maggiori informazioni.
- I gruppi di cattura possono essere usati all'interno dell'espressione regolare stessa, utilizzando rispettivamente *\N* al posto di *%N*. Per maggiori informazioni, vedi [catturare il testo corrispondente \(riferimenti all'indietro\)](#) nelle [espressioni regolari](#).

Esempio 1:

In questo semplice esempio il testo corrispondente all'espressione regolare **=*** viene catturato ed inserito in *%1* nella regola dinamica. Ciò permette al commento di terminare con la stessa quantità di **=** che ha all'inizio. Questo associa del testo tipo: **[[commento]]**, **[=[commento]=]** oppure **[====[commento]=====]**.

Inoltre le catture sono disponibili solo nel contesto cambiato *Multi-line Comment*.

```
<context name="Normal" attribute="Normal Text" lineEndContext="#stay">
  <RegExpr context="Multi-line Comment" attribute="Comment" String="\[ (=*) ←
    \[" beginRegion="RegionComment"/>
</context>
<context name="Multi-line Comment" attribute="Comment" lineEndContext="# ←
  stay">
  <StringDetect context="#pop" attribute="Comment" String="]%1]" dynamic=" ←
    true" endRegion="RegionComment"/>
</context>
```

Esempio 2:

Nella regola dinamica, *%1* corrisponde alle catture che associano **#+**, e *%2* a **"+"**. Ciò associa del testo come: **#label~~~~~dentro al contesto~~~~#**.

Queste catture non saranno disponibili in altri contesti, tipo *OtherContext*, *FindEscapes* oppure *SomeContext*.

```
<context name="SomeContext" attribute="Normal Text" lineEndContext="#stay">
  <Regex context="#pop!NamedString" attribute="String" String="(#+) (?:[\w ←
-]|[^[:ascii:]]) (&quot;+)" />
</context>
<context name="NamedString" attribute="String" lineEndContext="#stay">
  <Regex context="#pop!OtherContext" attribute="String" String="%2(?:%1) ←
?" dynamic="true" />
  <DetectChar context="FindEscapes" attribute="Escape" char="\\" />
</context>
```

Esempio 3:

Questo corrisponde a testo come: **Class::function<T>(...)**.

```
<context name="Normal" attribute="Normal Text" lineEndContext="#stay">
  <Regex context="FunctionName" lookAhead="true"
    String="\b([a-zA-Z_][\w-]*) (::) ([a-zA-Z_][\w-]*) (?:&lt;[\w-\ ←
s]*&gt;)? (\)" />
</context>
<context name="FunctionName" attribute="Normal Text" lineEndContext="#pop">
  <StringDetect context="#stay" attribute="Class" String="%1" dynamic="true ←
"/>
  <StringDetect context="#stay" attribute="Operator" String="%2" dynamic=" ←
true"/>
  <StringDetect context="#stay" attribute="Function" String="%3" dynamic=" ←
true"/>
  <DetectChar context="#pop" attribute="Normal Text" char="4" dynamic="true ←
"/>
</context>
```

DELIMITATORI LOCALI

- *weakDelimiter*: elenco di caratteri che non agiscono come delimitatori di parole.
- *additionalDelimiter*: definisce dei delimitatori addizionali.

6.2.4.1 Le regole in dettaglio

DetectChar

Rileva un singolo carattere specifico. Usato comunemente per trovare ad esempio la fine di una stringa tra virgolette.

```
<DetectChar char="(carattere)" (attributi comuni) (dynamic) />
```

L'attributo **char** definisce il carattere da abbinare.

Detect2Chars

Rileva due caratteri specifici in un ordine definito.

```
<Detect2Chars char="(carattere)" char1="(carattere)" (attributi comuni) ←
/>
```

L'attributo **char** definisce il primo carattere da associare, **char1** il secondo.

Questa regola è presente per ragioni storiche, e per leggibilità è preferibile usare **StringDetect**.

AnyChar

Rileva un carattere da un insieme di caratteri specificati.

```
<AnyChar String="(stringa)" (attributi comuni) />
```

L'attributo **String** definisce l'insieme dei caratteri.

StringDetect

Rileva una stringa esatta.

```
<StringDetect String="(stringa)" [insensitive="true|false"] (attributi comuni) (dynamic) />
```

L'attributo **String** definisce la stringa da abbinare. All'attributo **insensitive** viene assegnato come valore predefinito *false*, e viene passato alla funzione di confronto della stringa. Se il valore è *true* allora nel confronto non si terrebbe conto delle maiuscole.

WordDetect

Non rileva una stringa esatta, ma i confini di una parola quali un punto '.' o uno spazio all'inizio o alla fine della parola. Pensa `\b<string>\b` come un'espressione regolare, ma più veloce della regola **RegExpr**.

```
<WordDetect String="(stringa)" [insensitive="true|false"] (attributi comuni) (delimitatori locali) />
```

L'attributo **String** definisce la stringa da abbinare. All'attributo **insensitive** viene assegnato come valore predefinito *false*, e viene passato alla funzione di confronto della stringa. Se il valore è *true* allora nel confronto non si terrebbe conto delle maiuscole.

Da: Kate 3.5 (KDE 4.5)

RegExpr

Corrisponde con un'espressione regolare.

```
<RegExpr String="(stringa)" [insensitive="true|false"] [minimal="true|false"] (attributi comuni) (dynamic) />
```

L'attributo **String** definisce l'espressione regolare.

insensitive ha come valore predefinito *false*, e viene passato al motore per le espressioni regolari.

minimal ha come valore predefinito *false*, e viene passato al motore per le espressioni regolari.

Le regole sono sempre confrontate con l'inizio della stringa corrente, tuttavia un'espressione regolare iniziante con un apice (^) è segno che la regola dovrebbe essere confrontata invece con l'inizio di una riga.

Vedi [Espressioni regolari](#) per maggiori informazioni.

keyword

Rileva una parola chiave da una lista specifica.

```
<keyword String="(nome lista)" (attributi comuni) (delimitatori locali) />
```

L'attributo **String** identifica la lista di parole chiave per nome. Deve esistere una lista con questo nome.

Il sistema di evidenziazione elabora regole di parole chiave in maniera molto ottimizzata. Questo rende assolutamente necessario che ogni parola chiave da confrontare sia circondata da delimitatori definiti, sia impliciti (i delimitatori predefiniti) che espliciti, specificati dalla proprietà *additionalDelimiter* con l'etichetta *keywords*.

Se una parola chiave da confrontare dovesse contenere un carattere di delimitazione, allora questo dovrebbe essere aggiunto alla proprietà *weakDelimiter* con l'etichetta *keywords*. Questo carattere perderebbe le sue proprietà di delimitatore in tutte le regole *keywords*. È possibile anche usare l'attributo *weakDelimiter* di *keyword* in modo da applicare questa modifica solo a questa regola.

Int

Rileva un numero intero (come nell'espressione regolare `\b[0-9]+`).

```
<Int (attributi comuni) (delimitatori locali) />
```

Questa regola non ha attributi specifici.

Float

Rileva un numero in virgola mobile (come nell'espressione regolare `(\b[0-9]+\.[0-9]*|\.[0-9]+)([eE][-+]?[0-9]+)?`).

```
<Float (attributi comuni) (delimitatori locali) />
```

Questa regola non ha attributi specifici.

HiCOct

Rileva un numero rappresentato in ottale (come nell'espressione regolare `\b0[0-7]+`).

```
<HiCOct (attributi comuni) (delimitatori locali) />
```

Questa regola non ha attributi specifici.

HiCHex

Rileva un numero rappresentato in esadecimale (come nell'espressione regolare `\b0[xX][0-9a-fA-F]+`).

```
<HiCHex (attributi comuni) (delimitatori locali) />
```

Questa regola non ha attributi specifici.

HiCStringChar

Rileva un carattere di escape.

```
<HiCStringChar (attributi comuni) />
```

Questa regola non ha attributi specifici.

Controlla la corrispondenza di espressioni letterali di caratteri generalmente usate nel codice dei programmi, per esempio `\n` (nuova riga) o `\t` (TAB).

I seguenti caratteri corrisponderanno se saranno seguiti da una barra inversa (`\`): **abefnrtv''?\'**. Inoltre corrisponderanno numeri esadecimali preceduti dal carattere di escape, come per esempio `\xff` e numeri ottali preceduti dal carattere di escape, per esempio `\033`.

HiCChar

Rileva il carattere C

```
<HiCChar (attributi comuni) />
```

Questa regola non ha attributi specifici.

Controlla la corrispondenza di caratteri C racchiusi da un accento (Esempio: `'c'`). Gli accenti possono essere caratteri singoli o caratteri di escape. Vedi `HiCStringChar` per le sequenze di caratteri di escape associate.

RangeDetect

Rileva una stringa con caratteri di inizio e di fine definiti.

```
<RangeDetect char="(carattere)" char1="(carattere)" (attributi comuni) ↔
/>
```

char definisce il carattere con cui inizia l'intervallo, **char1** il carattere con cui finisce.

Utile per individuare per esempio piccole stringhe tra virgolette e simili; nota però che non verranno in questo modo trovate le stringhe che si estendono dopo un'interruzione di riga. Questo perché il motore di evidenziazione lavora su una riga alla volta.

LineContinue

Associa un carattere specificato con la fine di una riga.

```
<LineContinue (attributi comuni) [char="\"] />
```

char carattere opzionale da associare; come predefinito c'è la barra inversa ('\''). Novità da KDE 4.13.

Questa regola è utile per cambiare il contesto alla fine di una riga. È necessaria per esempio in C/C++, per continuare le macro o le stringhe.

IncludeRules

Include delle regole da un altro file di contesto o da un linguaggio.

```
<IncludeRules context="contextlink" [includeAttrib="true|false"] />
```

L'attributo **context** definisce quale contesto includere.

Se è una stringa semplice include tutte le regole nel contesto corrente, per esempio:

```
<IncludeRules context="altroContesto" />
```

Se la stringa contiene **##** il sistema di evidenziazione cercherà un contesto in un'altra definizione di linguaggio con il nome assegnato, per esempio

```
<IncludeRules context="String##C++" />
```

dovrebbe includere il contesto *String* dalle definizioni di evidenziazione del C++.

Se l'attributo **includeAttrib** è *true* cambia l'attributo di destinazione con uno del sorgente. Questo è richiesto per esempio per fare dei commenti, se il testo associato al contesto incluso ha una diversa evidenziazione dal contesto che lo ospita.

DetectSpaces

Rileva gli spazi.

```
<DetectSpaces (attributi comuni) />
```

Questa regola non ha attributi specifici.

Usa questa regola quando sai che potrai incontrare numerosi spazi, per esempio all'inizio di righe rientrate. Questa regola salterà tutti gli spazi in un colpo solo, invece di provare regole multiple da scartarle di volta in volta per mancanza di corrispondenza.

DetectIdentifier

Rileva gli identificatori di stringhe (come un'espressione regolare **[a-zA-Z_][a-zA-Z0-9_]***).

```
<DetectIdentifier (attributi comuni) />
```

Questa regola non ha attributi specifici.

Usa questa regola per saltare subito una stringa di caratteri alfanumerici, invece di usare più regole e saltare i caratteri uno alla volta per mancanza di corrispondenza.

6.2.4.2 Suggerimenti e trucchi

- Le espressioni regolari sono semplici da usare, ma spesso c'è un altro modo molto più veloce per ottenere gli stessi risultati. Considera di voler semplicemente cercare la corrispondenza del carattere «#» solo quando è il primo della riga. Una soluzione basata su un'espressione regolare assomiglierebbe a questo:

```
<RegExp attribute="Macro" context="macro" String="^\s*#" />
```

Puoi ottenere lo stesso risultato in maniera più veloce usando:

```
<DetectChar attribute="Macro" context="macro" char="#" firstNonSpace="↵
true" />
```

Se vuoi cercare corrispondenze per l'espressione regolare '**^#**' puoi sempre usare **DetectChar** con l'attributo **column="0"**. L'attributo **column** conta i caratteri, e un tabulatore è un solo carattere.

- Nelle regole **RegExp** usa l'attributo **column="0"** se il modello **^MODELLO** verrà utilizzato per trovare il testo all'inizio della riga. Ciò migliora le prestazioni, perché evita di cercare le corrispondenze nel resto delle colonne.
- Nelle espressioni regolari usa i gruppi non di cattura (**? :MODELLO**) invece dei gruppi di cattura (**MODELLO**) se le catture non verranno utilizzate nella stessa espressione regolare o nelle regole dinamiche. Ciò evita di salvare delle catture quando non è necessario.
- Puoi cambiare i contesti senza bisogno di elaborare dei caratteri. Se vuoi cambiare contesto quando incontri la stringa ***/** devi almeno elaborare quella stringa nel contesto del testo. La regola qui sotto corrisponderà, e l'attributo **lookAhead** farà in modo che l'evidenziatore tenga la stringa corrispondente per il contesto successivo.

```
<StringDetect attribute="Comment" context="#pop" String="*/" lookAhead="↵
true" />
```

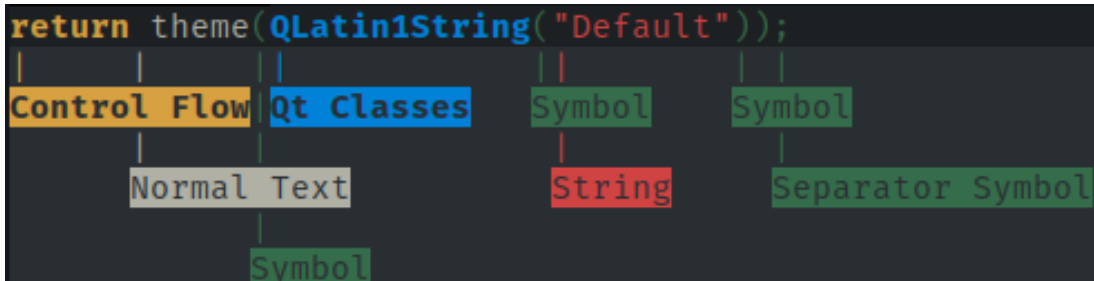
- Usa **DetectSpaces** se sai di incontrare molti spazi.
- Usa **DetectIdentifier** al posto dell'espressione regolare '**[a-zA-Z_]\w***'.
- Usa gli stili predefiniti ogni volta che puoi. In questo modo l'utente si troverà in un ambiente familiare.
- Guarda dentro altri file XML, per vedere come le altre persone implementano le regole difficili.
- Puoi validare ogni file XML usando il comando **validatehl.sh mySyntax.xml**. Il file **validatehl.sh** usa **language.xsd**, che sono entrambi disponibili nel [deposito di evidenziazione della sintassi](#).
- Se ripeti molto spesso delle espressioni regolari complesse puoi usare **ENTITIES**. Esempio:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE language
[
    <!ENTITY myref      "[A-Za-z_:] [\w.:-]*">
]>
```

Ora puoi usare *&myref;* invece dell'espressione regolare.

- Nell'editor Kate puoi ricaricare le sintassi usando la riga di comando incorporata (**F7** è la scorciatoia per impostazione predefinita) e il comando **reload-highlighting**.

- Puoi usare il programma da riga di comando chiamato **ksyntaxhighlighter6** (**kate-syntax-highlighter** nelle versioni più vecchie) per provare una sintassi e per visualizzare lo stile e le regioni associate con ciascuna parte di un testo.



Il risultato di `ksyntaxhighlighter6 --output-format=ansi --syntax-trace=format test.cpp`.

Usa `ksyntaxhighlighter6 -h` per le altre opzioni.

6.3 Lavorare con i temi di colori

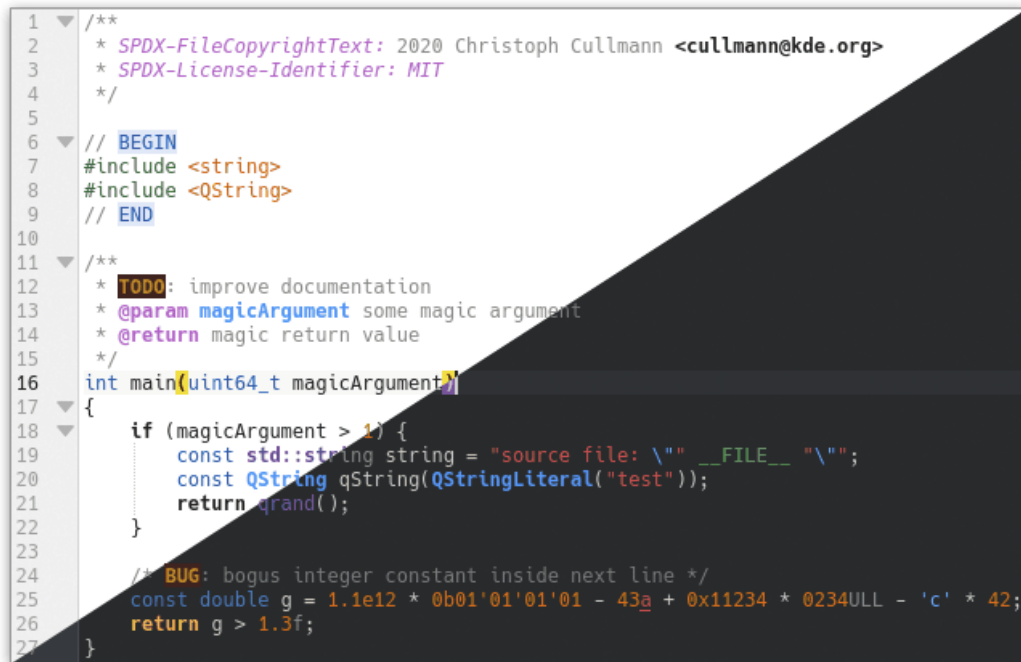
6.3.1 Panoramica

I temi di colori definiscono i colori dell'[area di modifica del testo](#) e dell'[evidenziazione della sintassi](#). Un tema di colori comprende quanto segue:

- Lo stile del testo, usato per l'evidenziazione della sintassi attraverso gli *attributi di stile predefiniti*, ad esempio il colore del testo e di quello del testo selezionato.
- Lo sfondo dell'area di modifica del testo, inclusa la selezione del testo e della riga corrente.
- Il bordo dell'icona dell'area del testo: il loro sfondo, la riga di separazione, i numeri di riga, i marcatori di ritorno a capo della riga, i segni per le righe modificate e di raggruppamento del codice.
- I decorator di testo, come i marcatori di ricerca e quelli di rientro e di tabulazione o di spazio, la corrispondenza delle parentesi e il correttore ortografico.
- Segnalibri e frammenti.

Per evitare confusione, quanto segue è fuori ambito:

- Il tipo di carattere e la sua dimensione.
- I colori dell'applicazione di modifica del testo, ad esempio quelli della mappa nella barra di scorrimento, del menu, della barra del titolo della finestra, eccetera. Nelle applicazioni di KDE, quali Kate o KDevelop, questi colori sono definiti dallo **schema di colori globali di KDE Plasma**, che si trova nel modulo [Colori delle Impostazioni di sistema](#) o nell'applicazione stessa, nel menu **Impostazioni** → **Schema di colori**.



```

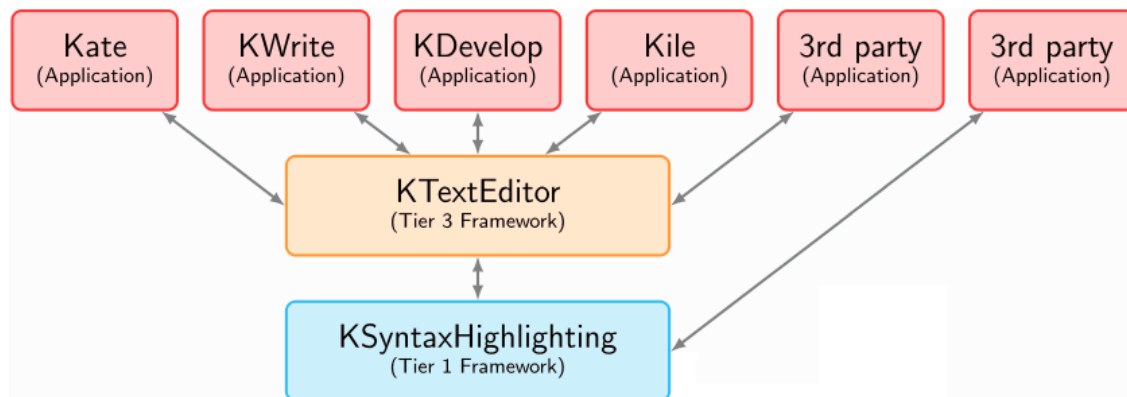
1  /**
2   * SPDX-FileCopyrightText: 2020 Christoph Cullmann <cullmann@kde.org>
3   * SPDX-License-Identifier: MIT
4   */
5
6  // BEGIN
7  #include <string>
8  #include <QString>
9  // END
10
11 /**
12  * TODO: improve documentation
13  * @param magicArgument some magic argument
14  * @return magic return value
15  */
16 int main(uint64_t magicArgument)
17 {
18     if (magicArgument > 1) {
19         const std::string string = "source file: \"__FILE__\"";
20         const QString qString(QStringLiteral("test"));
21         return qrand();
22     }
23
24     /* BUG: bogus integer constant inside next line */
25     const double g = 1.1e12 * 0b01'01'01'01 - 43a + 0x11234 * 0234ULL - 'c' * 42;
26     return g > 1.3f;
27 }

```

I temi di colori 'Brezza chiaro' e 'Brezza scuro' con l'evidenziazione della sintassi del 'C++'.

6.3.2 I temi di colori di KSyntaxHighlighting

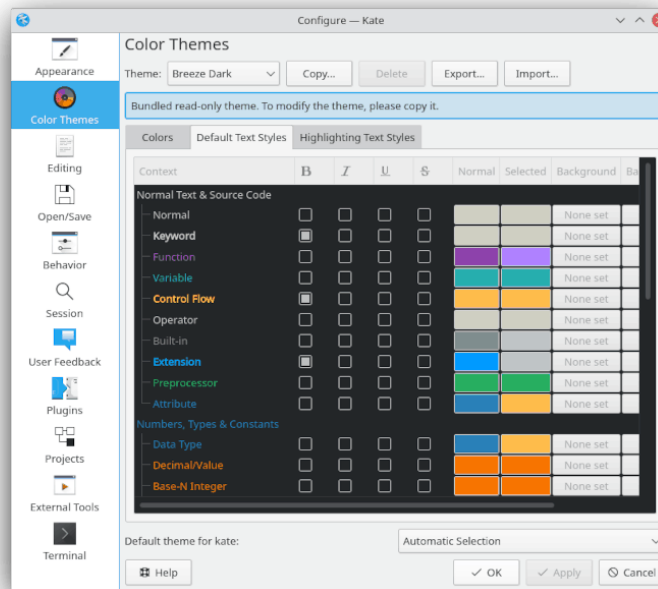
Il framework [KSyntaxHighlighting](#), che è il motore dell'[evidenziazione della sintassi](#), è la libreria che **fornisce e gestisce i temi di colori**. È parte di KDE Frameworks e viene usata negli editor di testo di KDE come [Kate](#), [KWrite](#), [Kile](#) e [KDevelop](#). Questa dipendenza assomiglia a quanto segue:



Dipendenze delle librerie di KDE Frameworks negli editor di testo.

[KSyntaxHighlighting](#) include una varietà di temi incorporati che vengono visualizzati nella [pagina temi di colori sul sito web dell'editor Kate](#).

Il framework [KTextEditor](#), che è il motore di modifica dei testi, fornisce un'interfaccia utente per creare e per modificare i temi di colore, incluso uno strumento per importarli e per esportarli. Questo è il modo più semplice di creare e di modificare i temi, a cui puoi accedere dalla [finestra 'Configura'](#) dell'editor. Maggiori dettagli in Sezione [6.3.5](#).



L'GUI per gestire i temi di colori nelle impostazioni di Kate.

È importante menzionare che negli editor di testo di KDE, come Kate o KDevelop, i temi di colori di KSyntaxHighlighting vengono usati [da KDE Frameworks 5.75](#), rilasciato il 10 ottobre 2020. In precedenza venivano usati gli schemi di colori di Kate (schemi di configurazione basati su KConfig) che ora sono deprecati. Ad ogni modo è possibile convertire i vecchi schemi di Kate nei temi di colori di KSyntaxHighlighting: il [deposito di KSyntaxHighlighting](#) include a questo scopo lo script `utils/kateschema_to_theme_converter.py` e il programma `utils/schema-converter/`.

6.3.3 Il formato JSON dei temi di colori

6.3.3.1 Panoramica

I temi di colori sono memorizzati in file in formato JSON con l'estensione `.theme`.

Nel [codice sorgente di KSyntaxHighlighting](#) i file JSON dei temi incorporati sono nella cartella `data/themes/`. Nota che negli editor di testo i temi incorporati sono compilati nella libreria KSyntaxHighlighting, quindi il modo per accedervi è tramite il codice sorgente oppure esportandoli dalla GUI che gestisce i temi di KTextEditor.

È anche possibile aggiungere facilmente dei temi aggiuntivi o personalizzati, che vengono caricati dal file system. I file dei temi personalizzati dall'utente si trovano nella cartella `org.kde.syntax-highlighting/themes/` che si trova nella cartella utente e vengono trovati col comando `qtpaths --paths GenericDataLocation`; di solito sono `$HOME/.local/share/` e `/usr/share/`.

Per i pacchetti Flatpak e Snap la cartella qui sopra non funzionerà perché la posizione dei dati è diversa per ogni applicazione. In un'applicazione Flatpak generalmente la posizione dei file dei temi personalizzati è `$HOME/.var/app/nome-del-pacchetto-flatpak/data/org.kde.syntax-highlighting/themes/`, mentre in una Snap è `$HOME/.snap/nome-del-pacchetto-snap/current/.local/share/org.kde.syntax-highlighting/themes/`.

In Windows® questi file si trovano in `%USERPROFILE%\AppData\Local\org.kde.syntax-highlighting\themes`. `%USERPROFILE%` viene generalmente espanso in `C:\Users\utente`.

In sintesi, per la maggior parte delle configurazioni la cartella dei temi personalizzati è la seguente:

Per l'utente locale	\$HOME /.local/share/org.kde.syntax-highlighting/themes/
Per tutti gli utenti	/usr/share/org.kde.syntax-highlighting/themes/
Per i pacchetti Flatpak	\$HOME /.var/app/nome-pacchetto-flatpak /data/org.kde.syntax-highlighting/themes/
Per i pacchetti Snap	\$HOME /snap/nome-pacchetto-snap /current/.local/share/org.kde.syntax-highlighting/themes/
In Windows®	%USERPROFILE%\AppData\Local\org.kde.syntax-highlighting/themes
In macOS®	\$HOME /Library/Application Support/org.kde.syntax-highlighting/themes/

Se esistono più temi con lo stesso nome, verrà caricato quello che ha il valore più alto dell'attributo **revision**.

6.3.3.2 La struttura JSON

La struttura di un file JSON è spiegata nel [loro sito web](#). Fondamentalmente il formato di file JSON consiste in:

- Collezioni di coppie di chiavi e valori separate da virgole e raggruppate tra { }, che chiameremo 'oggetti'.
- Elenchi ordinati di valori separati da virgole e raggruppate tra [], che chiameremo 'vettori'.

In questo articolo verrà usata la nomenclatura 'chiave', 'valore', 'oggetto' e 'vettore'. Se è la prima volta che lavori con i file JSON la loro comprensione è semplice come guardare l'esempio qui sotto,

6.3.3.3 Struttura principale dei file JSON di un tema di colori

L'oggetto principale del file del tema di colori JSON contiene il seguente schema di chiavi:

- **metadata**: è obbligatorio. Il valore è un oggetto coi metadati del tema, come il nome, la revisione e la licenza.
È spiegato nei dettagli in Sezione [6.3.3.4](#).
- **editor-colors**: è obbligatorio. Il valore è un oggetto con i colori dell'area di modifica del testo, come lo sfondo il bordo dell'icona e le decorazioni del testo.
È spiegato nei dettagli in Sezione [6.3.4.1](#).
- **text-styles**: è obbligatorio. Il valore è un oggetto con gli attributi *stili di testo predefiniti* dell'evidenziazione della sintassi. Ciascun attributo definisce il suo *colore del testo*, *colore del testo selezionato*, o se, ad esempio, è in *grassetto* o in *corsivo*. Gli stili del testo possono essere referenziati dagli [attributi dei file XML di definizione della sintassi](#).
È spiegato nei dettagli in Sezione [6.3.4.2](#).

- **custom-styles**: è facoltativo. Definisce gli stili di testo per gli attributi di definizioni di evidenziazione della sintassi specifiche. Ad esempio, in una definizione di evidenziazione come **Python** o **Markdown** puoi specificare uno stile di testo diverso che sostituisce quello predefinito che è definito in **text-styles**.

È spiegato nei dettagli in Sezione [6.3.4.3](#).

Il linguaggio JSON non supporta i commenti, tuttavia puoi usare la chiave facoltativa **_comments** nell'oggetto principale per scrivere dei commenti; ad esempio, se stai adattando un tema esistente puoi mettere l'URL del deposito. Il modo più pratico è quello di usare un vettore di stringhe.

Qui sotto c'è un file di esempio del tema 'Brezza chiaro'. Puoi notare che, per evitare che l'esempio sia troppo lungo, gli oggetti **editor-colors** e **text-styles**, non contengano tutte le chiavi richieste. Puoi vedere l'intero archivio del [tema Brezza chiaro nel deposito di KSyntaxHighlighting](#).

```
{
  "_comments": [
    "Questo è un commento.",
    "Se questo tema è un adattamento di un altro, inserisci il ↔
      collegamento alla sorgente d'installazione originale."
  ],
  "metadata": {
    "name" : "Breeze Light",
    "revision" : 5,
    "copyright": [
      "SPDX-FileCopyrightText: 2016 Volker Krause <vkrause@kde.org>",
      "SPDX-FileCopyrightText: 2016 Dominik Haumann <dhaumann@kde.org ↔
        >"
    ],
    "license": "SPDX-License-Identifier: MIT"
  },
  "editor-colors": {
    "BackgroundColor" : "#ffffff",
    "CodeFolding" : "#94caef",
    "BracketMatching" : "#ffff00",
    "CurrentLine" : "#f8f7f6",
    "IconBorder" : "#f0f0f0",
    "IndentationLine" : "#d2d2d2",
    "LineNumbers" : "#a0a0a0",
    "CurrentLineNumber" : "#1e1e1e",
    "Le altre chiavi dei colori dell'editor...",
  },
  "text-styles": {
    "Normal" : {
      "text-color" : "#1f1c1b",
      "selected-text-color" : "#ffffff",
      "bold" : false,
      "italic" : false,
      "underline" : false,
      "strike-through" : false
    },
    "Keyword" : {
      "text-color" : "#1f1c1b",
      "selected-text-color" : "#ffffff",
      "bold" : true
    },
    "Function" : {
```

```

        "text-color" : "#644a9b",
        "selected-text-color" : "#452886"
    },
    "Variable" : {
        "text-color" : "#0057ae",
        "selected-text-color" : "#00316e"
    },
    Le altri chiavi di stile del testo...

},
"custom-styles": {
    "ISO C++": {
        "Data Type": {
            "bold": true,
            "selected-text-color": "#009183",
            "text-color": "#00b5cf"
        },
        "Keyword": {
            "text-color": "#6431b3"
        }
    },
    "YAML": {
        "Attribute": {
            "selected-text-color": "#00b5cf",
            "text-color": "#00b5cf"
        }
    }
}
}

```

6.3.3.4 Metadati

L'oggetto JSON della chiave **metadata** contiene informazioni rilevanti sul tema. Questo oggetto ha le seguenti chiavi:

- **name**: è una *stringa* impostata al nome del linguaggio. Appare successivamente nei menu e nelle finestre. È obbligatoria.
- **revision**: è un numero *integer* che specifica l'attuale revisione del file del tema. Assicurati di incrementare questo numero ogni volta che aggiorni un file di tema di colori. È obbligatoria.
- **license**: è una *stringa* che definisce la licenza del tema usando l'identificativo **SPDX-License-Identifier** dello standard del [formato di comunicazione della licenza SPDX](#). È facoltativa.
[Qui puoi vedere l'elenco completo degli identificatori di licenza SPDX.](#)
- **copyright**: è un *vettore* di *stringhe* che specifica l'autore del tema usando l'identificativo **SPDX-FileCopyrightText** dello standard del [formato di comunicazione della licenza SPDX](#). È facoltativa.

```

"metadata": {
    "name" : "Brezza chiaro",
    "revision" : 5,
    "copyright": [
        "SPDX-FileCopyrightText: 2016 Volker Krause <vkrause@kde.org>",
        "SPDX-FileCopyrightText: 2016 Dominik Haumann <dhaumann@kde.org>"
    ],

```

```
"license": "SPDX-License-Identifier: MIT"
}
```

6.3.4 I colori in dettaglio

Questa sezione descrive in dettaglio tutti gli attributi di colore disponibili e le tutte le impostazioni di colore disponibili.

6.3.4.1 Modificare i colori

Corrisponde ai colori dell'[area di modifica del testo](#).

Nel [file di tema JSON](#) la rispettiva chiave **editor-colors** ha come valore un *oggetto* nel quale ciascuna chiave fa riferimento ad un attributo di colore dell'editor di testo. Qui **tutte le chiavi disponibili sono obbligatorie**: i loro valori sono delle **stringhe** con i codici esadecimali dei colori, ad esempio '#00B5CF'.

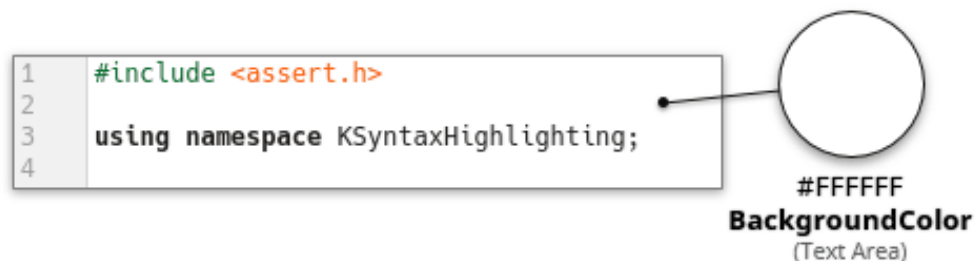
Nell'[GUI per gestire i temi di KTextEditor](#) questi attributi possono essere modificati nella scheda **Colori**.

Le chiavi disponibili sono le seguenti: quelle usate nel [file JSON](#) sono elencate in *grassetto*, mentre i nomi usati nell'[GUI](#) vengono mostrati tra parentesi.

Colori di sfondo dell'editor

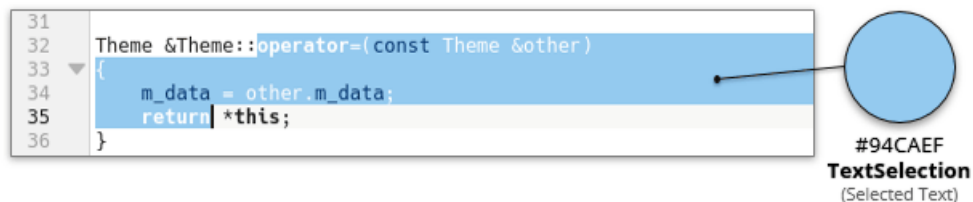
BackgroundColor (Area di testo)

Questo è lo sfondo predefinito per l'area dell'editor, sarà il colore dominante dell'area principale.



TextSelection (Testo selezionato)

Questo è lo sfondo del testo selezionato.



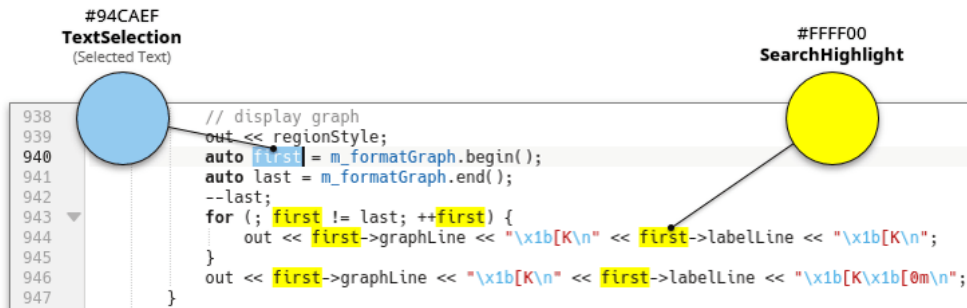
CurrentLine (Riga attuale)

Imposta il colore della riga attuale. Se è un po' diverso da quello del testo normale, permette di mantenere attiva la riga.



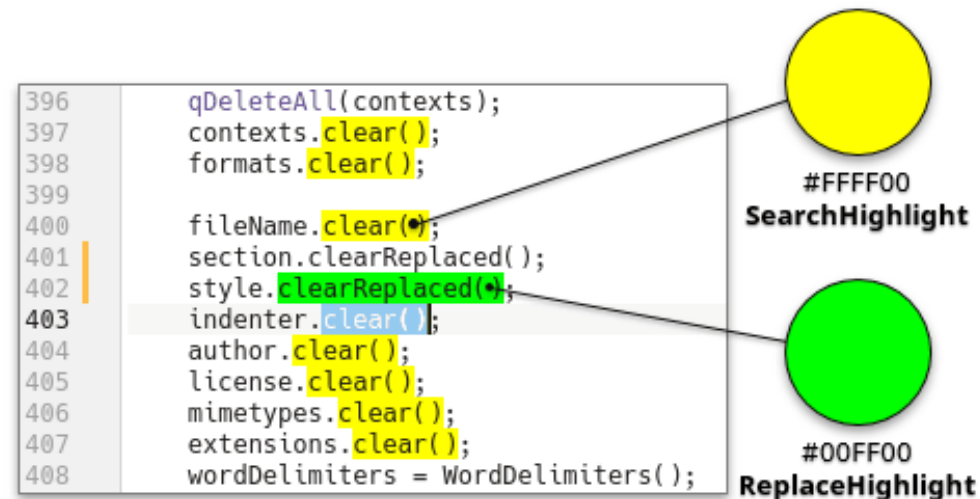
SearchHighlight (Evidenziazione delle ricerche)

Imposta il colore del testo che corrisponde all'ultima ricerca.



ReplaceHighlight (Evidenziazione delle sostituzioni)

Imposta il colore del testo che corrisponde all'ultima operazione di sostituzione.



Bordo delle icone

IconBorder (Area di sfondo)

Questo colore viene usato per gli indicatori, i bordi per i numeri di riga e per gli indicatori di raggruppamento sul lato sinistro della vista dell'editor, quando sono mostrati.

LineNumbers (Numeri di riga)

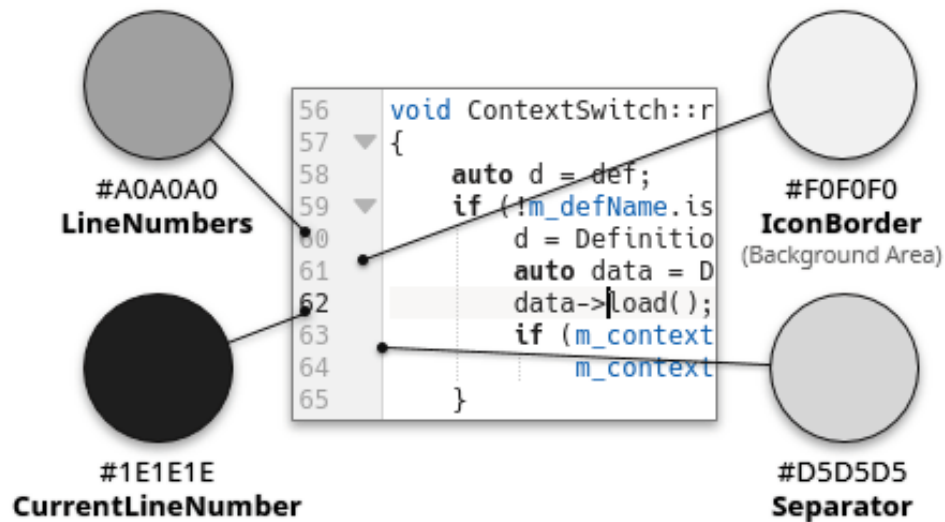
Questo colore viene usato per disegnare i numeri di riga sul lato sinistro della vista, quando sono mostrati.

CurrentLineNumber (Numero della riga attuale)

Questo colore viene usato per disegnare il numero della riga attuale, nella parte sinistra della vista quando è visualizzato. Impostandolo leggermente diverso da 'LineNumbers' può aiutare a mantenere attiva la riga attuale.

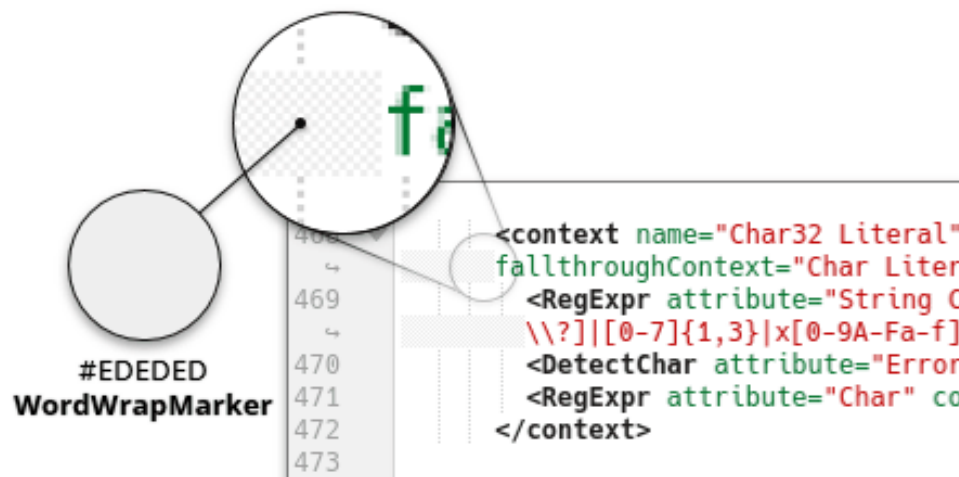
Separator (Separatore)

Questo colore viene usato per disegnare una linea verticale che separa il bordo dell'icona dallo sfondo dell'area del testo.



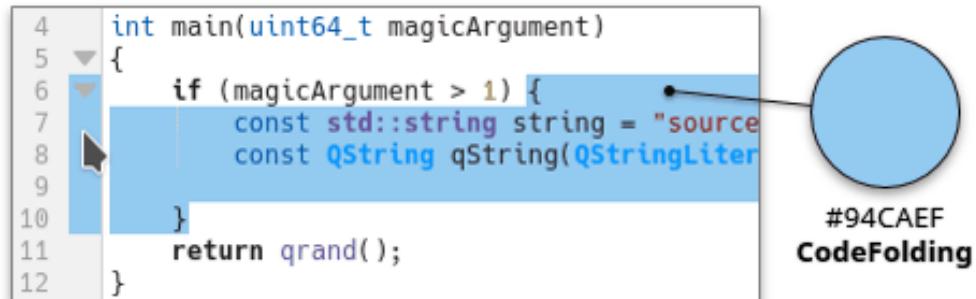
WordWrapMarker (Segno di ritorno a capo)

Questo colore viene usato per disegnare un motivo sulla sinistra delle righe fatte andare a capo automaticamente quando vengono allineate verticalmente, ed anche per l'indicatore di a capo statico.



CodeFolding (Raggruppamento del codice)

Questo colore viene usato per evidenziare la sezione di codice che sarebbe raggruppata al clic sulla freccia di raggruppamento a sinistra del documento. Per maggiori informazioni, vedi [la documentazione sul raggruppamento del codice](#).



ModifiedLines (Righe modificate)

Questo colore viene usato per evidenziare, alla sinistra di un documento, le righe che sono state modificate ma non ancora salvate. Per maggiori informazioni, vedi Sezione [3.9](#).

SavedLines (Righe salvate)

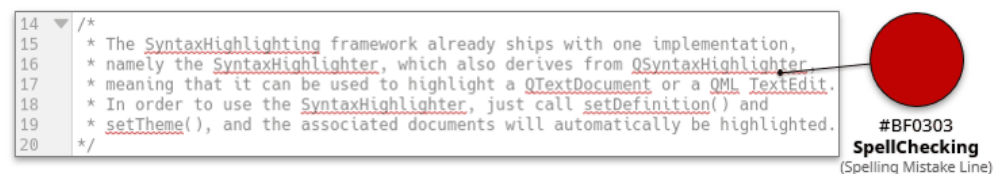
Questo colore viene usato per evidenziare indicare, alla sinistra di un documento, le righe che sono state modificate e salvate in questa sessione. Per maggiori informazioni, vedi Sezione [3.9](#).



Decorazioni del testo

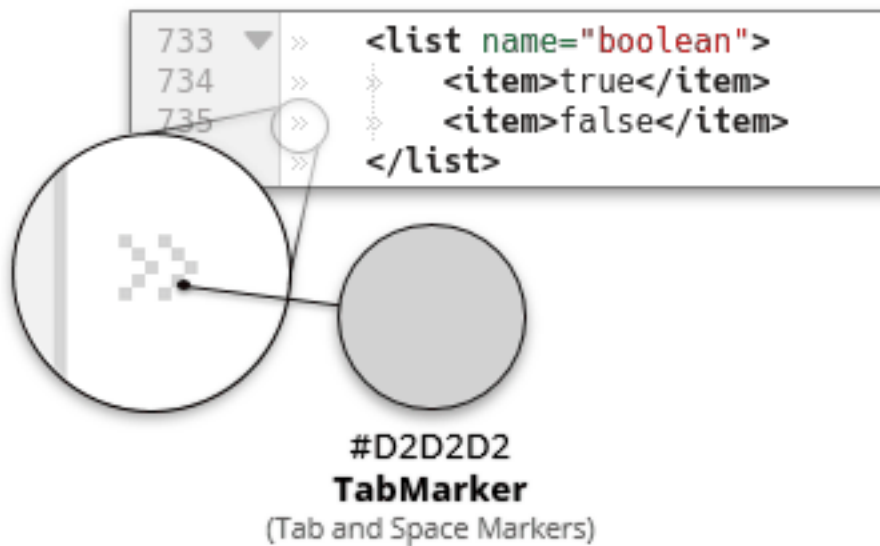
SpellChecking (Linea degli errori di ortografia)

Questo colore viene usato per indicare gli errori di ortografia.



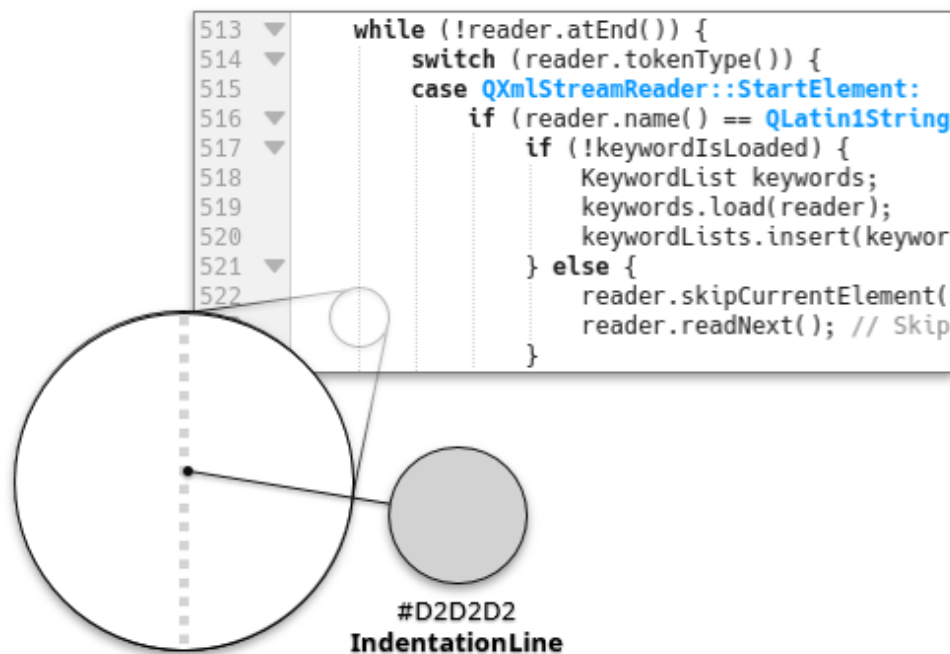
TabMarker (Segni di tabulazione e spazio)

Questo colore viene usato per gli indicatori di spazi bianchi, quando sono abilitati.



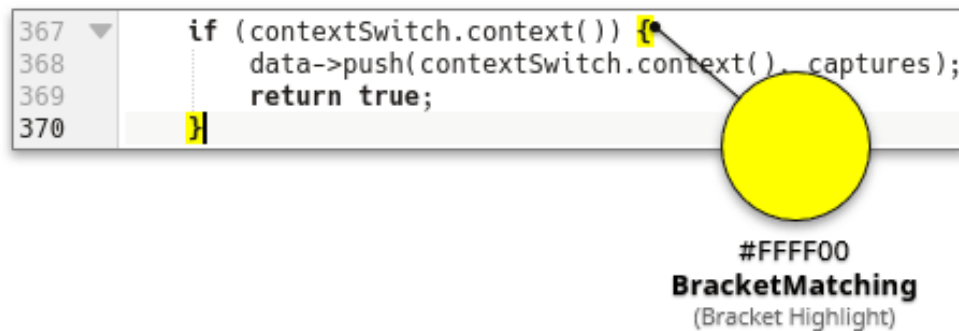
IndentationLine (Linea di rientro)

Questo colore viene usato per disegnare una linea alla sinistra dei blocchi rientrati, se questa [funzionalità è abilitata](#).



BracketMatching (Evidenziazione delle parentesi)

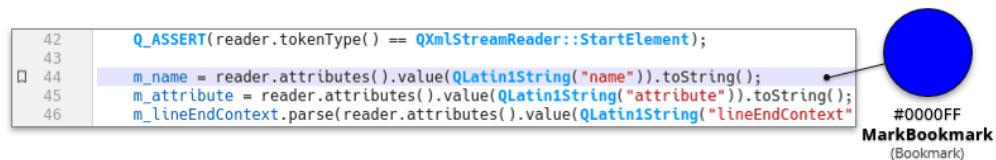
Questo colore viene usato per lo sfondo delle parentesi corrispondenti.



Colori degli indicatori

MarkBookmark (Segnalibro)

Questo colore viene usato per indicare i segnalibri. Nota che ha un'opacità del 22% (e del 33% per la riga attuale) rispetto allo sfondo. Per maggiori informazioni, vedi Sezione 3.6.



MarkBreakpointActive (Punto d'interruzione attivo)

Questo colore viene usato dall'estensione GDB per indicare un punto d'interruzione attivo. Nota che questo colore ha un'opacità sullo sfondo. Per maggiori informazioni, vedi [la documentazione dell'estensione GDB](#).

MarkBreakpointReached (Punto d'interruzione raggiunto)

Questo colore viene usato dall'estensione GDB per indicare un punto d'interruzione che hai raggiunto durante il debugging. Nota che questo colore ha un'opacità sullo sfondo. Per maggiori informazioni, vedi [la documentazione dell'estensione GDB](#).

MarkBreakpointDisabled (Punto d'interruzione disabilitato)

Questo colore viene usato dall'estensione GDB per indicare un punto d'interruzione inattivo. Nota che questo colore ha un'opacità sullo sfondo. Per maggiori informazioni, vedi [la documentazione dell'estensione GDB](#).

MarkExecution (Esecuzione)

Questo colore viene usato dall'estensione GDB per indicare la riga attualmente in esecuzione. Nota che questo colore ha un'opacità sullo sfondo. Per maggiori informazioni, vedi [la documentazione dell'estensione GDB](#).

MarkWarning (Avviso)

Questo colore viene usato dall'estensione di compilazione per indicare una riga che ha causato un avviso del compilatore. Nota che questo colore ha un'opacità sullo sfondo. Per maggiori informazioni, vedi [la documentazione dell'estensione di compilazione](#).

MarkError (Errore)

Questo colore viene usato dall'estensione di compilazione per indicare una riga che ha causato un errore del compilatore. Nota che questo colore ha un'opacità sullo sfondo. Per maggiori informazioni, vedi [la documentazione dell'estensione di compilazione](#).

Modelli e frammenti di testo

TemplateBackground (Sfondo)

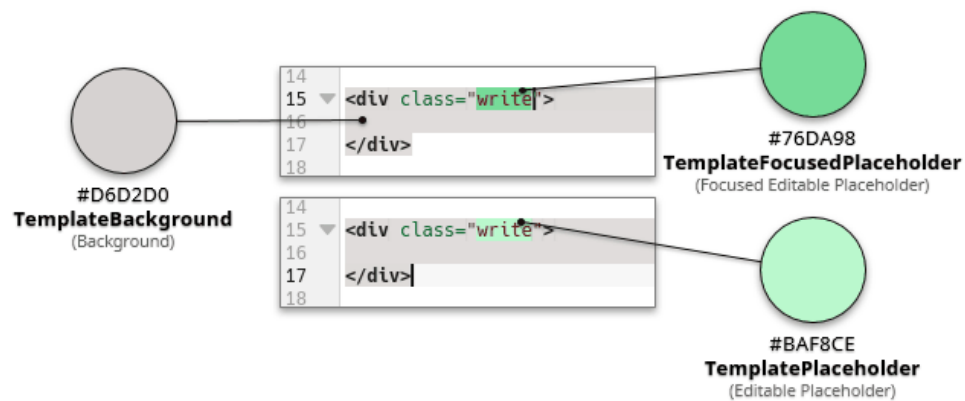
Questo colore è usato dall'estensione dei frammenti di Kate per colorare lo sfondo di un frammento. Per maggiori informazioni, vedi [la documentazione dei frammenti di Kate](#).

TemplatePlaceholder (Segnaposto modificabile)

Questo colore viene usato dall'estensione dei frammenti di Kate per marcare un segnaposto su cui puoi fare clic per modificarlo manualmente. Per maggiori informazioni, vedi [la documentazione dei frammenti di Kate](#).

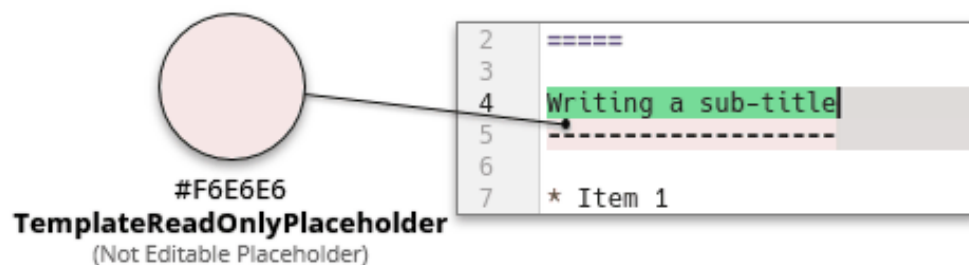
TemplateFocusedPlaceholder (Segnaposto modificabile attivo)

Questo colore viene usato dall'estensione dei frammenti di Kate per marcare il segnaposto che stai attualmente modificando. Per maggiori informazioni, vedi [la documentazione dei frammenti di Kate](#).



TemplateReadOnlyPlaceholder (Segnaposto non modificabile)

Questo colore viene usato dall'estensione dei frammenti di Kate per segnare un segnaposto non modificabile manualmente, per esempio uno che viene determinato automaticamente. Per maggiori informazioni, vedi [la documentazione dei frammenti di Kate](#).



6.3.4.2 Stili di testo predefiniti

Gli stili di testo predefiniti sono ereditati dagli stili di evidenziazione del testo, permettendo all'editor di presentare il testo in modo coerente; ad esempio, il testo dei commenti è uguale in quasi tutti i formati di testo che KSyntaxHighlighting può evidenziare.

NOTA

È possibile fare riferimento a questi **stili di testo** dagli stili predefiniti utilizzati nei file XML di definizione dell'[evidenziazione della sintassi](#). Ad esempio, l'attributo 'Normal' è equivalente a 'dsNormal' nei file XML, mentre 'DataType' a 'dsDataType'. Vedi Sezione [6.2.3.3](#) nella documentazione di evidenziazione della sintassi.

SUGGERIMENTO

Assicurati di scegliere dei colori leggibili e con un buon contrasto, specialmente in combinazione con l'**editor di colori**. Vedi Sezione [6.3.6.1](#).

Nel [file JSON](#) la rispettiva chiave **text-styles** ha come valore un *oggetto* nel quale ogni chiave corrisponde al nome di uno *stile di testo predefinito*, che sono equivalenti a quelli usati nelle definizioni di evidenziazione della sintassi. Qui **tutti le chiavi dello stile del testo disponibili sono obbligatorie** ; queste sono elencate di seguito.

```
"text-styles": {
  "Normal" : {
    "text-color" : "#1f1c1b",
    "selected-text-color" : "#ffffff",
    "bold" : false,
    "italic" : false,
    "underline" : false,
    "strike-through" : false
  },
  "Keyword" : {
    "text-color" : "#1f1c1b",
    "selected-text-color" : "#ffffff",
    "bold" : true
  },
  "Function" : {
    "text-color" : "#644a9b",
    "selected-text-color" : "#452886"
  },
  Le altre chiavi di stile del testo...
}
```

Ciascuna chiave degli *stili di testo predefiniti* ha un *oggetto JSON* per valore, dove vengono specificati valori come *colore*, *grassetto*, *corsivo*, ecc. Queste chiavi sono le seguenti:

text-color: è una *stringa* col colore del testo espresso come codice di colore esadecimale. Questa coppia di chiave e valore è richiesta.

selected-text-color: il colore del testo quando è selezionato; generalmente ha lo stesso valore di 'text-color'. Quando il testo viene selezionato, lo sfondo viene definito dal valore di [TextSelection](#) nell'**editor di colori**, quindi devi essere sicuro che il testo abbia un buon contrasto e che sia leggibile su questo sfondo. Questo valore è una *stringa* con un codice di colore esadecimale. Questa coppia di chiave e valore è richiesta.

bold: è un valore *booleano* che determina se il testo è in grassetto. Questa chiave è facoltativa, il suo valore predefinito è **false**.

italic: è un valore *booleano* che determina se il testo è curvo. Questa chiave è facoltativa, il suo valore predefinito è **false**.

underline: è un valore *booleano* che determina se il testo è sottolineato. Questa chiave è facoltativa, il suo valore predefinito è **false**.

strike-through: è un valore *booleano* che determina se il testo è barrato. Questa chiave è facoltativa, il suo valore predefinito è **false**.

background-color: determina lo sfondo del testo, che viene usato per esempio negli avvisi e nei commenti. Il valore è una *stringa* con un codice di colore esadecimale. Questa chiave è facoltativa, per impostazione predefinita non c'è uno sfondo.

selected-background-color: determina lo sfondo del testo quando viene selezionato. Il valore è una *stringa* con un codice di colore esadecimale. Questa chiave è facoltativa, per impostazione predefinita non c'è uno sfondo.

Nell'GUI per gestire i temi di colori di `KTextEditor` questi attributi possono essere modificati nella scheda **Stili di testo predefiniti**. Il nome nell'elenco degli stili utilizza lo stile configurato per l'elemento, fornendoti un'anteprima immediata durante la configurazione di uno stile. Ogni stile ti permette di selezionare gli attributi comuni, nonché i colori di primo piano e di sfondo. Per annullare l'impostazione di un colore di sfondo, fai clic con il pulsante destro per utilizzare il menu contestuale.

Le chiavi per lo stile del testo disponibili sono le seguenti: quelle usate nel [file JSON](#) sono elencate in *grassetto*, mentre i nomi usati nell'GUI vengono mostrati tra parentesi se sono diversi.

Testo normale e codice sorgente

Normal: lo stile del testo predefinito per il testo normale e per il codice sorgente senza un'evidenziazione speciale.

Keyword: lo stile del testo per le parole chiave incorporate nel linguaggio.

Function: lo stile del testo per le definizioni di funzioni e per le chiamate di funzioni.

Variable: lo stile del testo per le variabili, se applicabile. Per esempio, le variabili in PHP o in Perl iniziano tipicamente con \$, quindi tutti gli identificatori che seguono il modello `$foo` vengono evidenziati come variabili.

ControlFlow (Flusso di controllo): lo stile del testo per le parole chiave del flusso di controllo, come *if, then, else, return, switch, break, yield, continue*, ecc.

Operator: lo stile del testo per gli operatori, come `+, -, *, / , %`, ecc.

BuiltIn (Integrato): lo stile del testo per le classi del linguaggio, le funzioni e gli oggetti integrati.

Extension: lo stile del testo per le estensioni ben note, ad esempio le classi di Qt™, le funzioni e le macro del C++ e di Python o boost.

Preprocessor: lo stile del testo per le direttive al preprocessore o per le definizioni di macro.

Attribute: lo stile del testo per le annotazioni o per gli attributi delle funzioni o degli oggetti, ad esempio `@override` in Java, oppure `__declspec(...)` e `__attribute__(...)` in C++.

Numeri, tipi e costanti

DataType (Tipo di dato): stile del testo per i tipi di dati integrati, ad esempio *int, char, float, void, u64*, ecc.

DecVal (Decimale/Valore): stile del testo per i valori decimali.

BaseN (Intero in base N): stile del testo per i numeri in una base diversa da 10.

Float (Virgola mobile): stile del testo per i numeri in virgola mobile.

Constant: stile del testo per le costanti del linguaggio e per quelle definite dall'utente, ad esempio *True, False, None* in Python, oppure *nullptr* in C/C++, o anche per le costanti matematiche come *PI*.

Stringhe e caratteri

Char (Character): stile del testo per i singoli caratteri, ad esempio `'x'`.

SpecialChar (Carattere speciale): stile del testo per i caratteri di escape nelle stringhe, ad esempio `'hello\n'`, e per gli altri caratteri con un significato speciale nelle stringhe, ad esempio per gli operatori di sostituzione o di espressione regolare.

String: stile del testo per le stringhe, ad esempio `'hello world'`.

VerbatimString (Stringa letterale): stile del testo per le stringhe letterali o per quelle grezze, ad esempio `'raw \backslashbackslash'` in Perl, in CoffeeScript e in shells, così come `r' \raw'` in Python, o come negli *HERE docs*.

SpecialString (Stringa speciale): stile del testo per le stringhe speciali, ad esempio le espressioni regolari in ECMAScript, la modalità matematica di L^AT_EX, di SQL, ecc.

Import (Importazioni, moduli, inclusioni): stile del testo per le inclusioni, le importazioni, i moduli o i pacchetti di L^AT_EX.

Commenti e documentazione

Comment: lo stile del testo per i commenti normali.

Documentation: lo stile del testo per i commenti che riflettono la documentazione API, ad esempio `/** doxygen comments */` oppure `"""docstrings"""`.

Annotation: lo stile del testo per le annotazioni nei commenti o nei comandi della documentazione, ad esempio `@param` in Doxygen o in JavaDoc.

CommentVar (Variabile di commento): lo stile del testo che si riferisce ai nomi delle variabili usati nei comandi qui sopra in un commento, ad esempio `foobar` in `'@param foobar'`, in Doxygen o in JavaDoc.

RegionMarker (Delimitatore di regione): lo stile del testo per i delimitatori di regione, definiti tipicamente da `/ /BEGIN` e da `/ /END` nei commenti.

Information: lo stile del testo per le informazioni, per le note e per i suggerimenti, ad esempio la parola chiave `@note` in Doxygen.

Warning: lo stile del testo per gli avvisi, ad esempio la parola chiave `@warning` in Doxygen.

Alert: lo stile del testo per le parole speciali nei commenti, ad esempio `TODO`, `FIXME`, `XXXX` e `WARNING`.

Varie

Error: lo stile del testo per indicare l'evidenziazione degli errori e la sintassi errata.

Others: lo stile del testo per gli attributi che non soddisfano nessuno degli altri stili predefiniti.

6.3.4.3 Stili di evidenziazione del testo personalizzati

Qui puoi stabilire gli stili del testo per una definizione di evidenziazione della sintassi specifica, senza tener conto degli **stili di testo predefiniti** descritti nella [sezione precedente](#).

Nel [file di tema JSON](#) questo corrisponde alla chiave **custom-styles**, il cui valore è un *oggetto* nel quale ciascuna chiave del sottoschema corrisponde al **nome di una definizione di evidenziazione**. Il suo valore è un *oggetto* nel quale ciascuna chiave si riferisce al **nome degli attributi di stile** definito negli [elementi itemData](#) del file XML di evidenziazione della sintassi, e il rispettivo valore è un sotto-oggetto con le chiavi *text-color*, *selected-text-color*, *bold*, *italic*, *underline*, *strike-through*, *background-color* e *selected-background-color*, che sono state definite nella [sezione precedente](#). Ciascuno di questi valori è facoltativo, dal momento che viene considerato lo stile impostato in **text-styles** se non sono presenti.

Ad esempio, in questo pezzetto di codice la definizione di evidenziazione della sintassi di 'C++ ISO' ha uno stile del testo speciale per gli attributi 'Type Modifiers' e per 'Standard Classes'. Nel file XML corrispondente 'isocpp.xml', l'attributo definito 'Standard Classes' utilizza lo stile predefinito **BuiltIn** (o `dsBuiltIn`). In questo attributo, solamente il valore di **text-color** è sovrascritto dal nuovo colore '#6431b3'.

```
"custom-styles": {
  "ISO C++": {
    "Standard Classes": {
      "text-color": "#6431b3"
    },
    "Type Modifiers": {
      "bold": true,
      "selected-text-color": "#009183",
      "text-color": "#00b5cf"
    }
  }
}
```

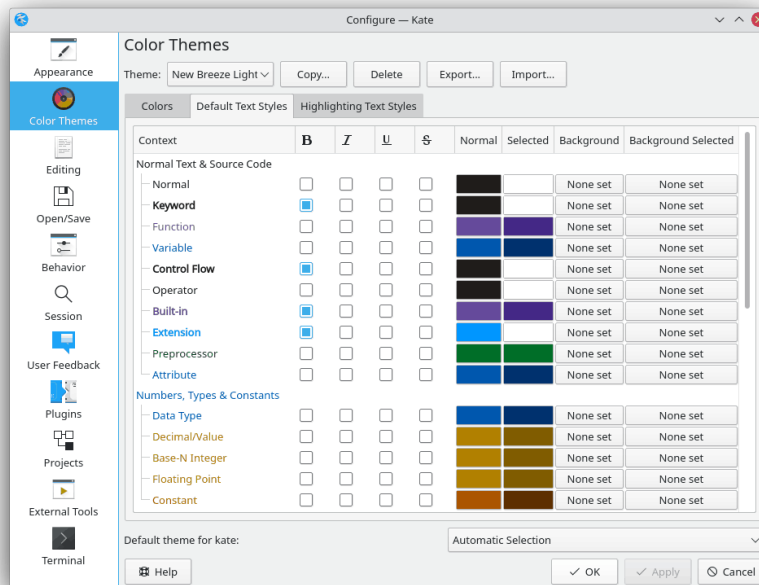
NOTA

- Dovresti considerare che questi stili di testo sono associati con i nomi degli attributi definiti nei file XML di evidenziazione della sintassi. Se un file XML viene aggiornato ed alcuni attributi vengono rinominati o rimossi, lo stile personalizzato definito nel tema non verrà più applicato.
- Le definizioni di evidenziazione della sintassi ne includono spesso delle altre. Ad esempio, l'evidenziatore di 'QML' include quello di 'JavaScript', dal momento che essi condividono delle funzionalità di evidenziazione.

Nell'GUI per gestire i temi di `KTextEditor` questi attributi possono essere modificati nella scheda **Stili di testo evidenziato**. Per impostazione predefinita, l'editor preseleziona l'evidenziazione del documento corrente. Noterai che molti evidenziatori ne contengono altri, rappresentati dai gruppi nell'elenco degli stili. Ad esempio, la maggior parte degli evidenziatori importa 'Alert', mentre molti dei formati di codice sorgente importano 'Doxygen'.

6.3.5 L'GUI temi di colori

Il modo più semplice per creare e per modificare i temi di colori è dall'GUI all'interno della **finestra 'Configura'** fornita da `KTextEditor`. Per accedervi, seleziona **Impostazioni → Configura Applicazione...** dalla barra del menu del tuo editor di testo. Questo fa apparire la finestra di dialogo **Configura**, lì seleziona **Temi di colore**, che è nel pannello laterale.



La finestra delle impostazioni di Kate con la gestione del tema dei colori.

In questa **finestra** puoi configurare tutti i colori di ogni tema tu abbia, così come creare o copiare nuovi temi, eliminarli, esportarli in un file **.theme** in **formato JSON**, o anche importarli da file **.theme** esterni. Ogni tema ha le impostazioni per i colori e per gli stili del testo.

I temi integrati non possono essere modificati per impostazione predefinita. Per fare ciò è necessario copiarli dandogli un nuovo nome.

Per usare permanentemente un tema nell'editor di testo lo devi selezionare nella casella combinata etichettata **Tema predefinito per Applicazione** in basso nella finestra, poi devi premere

Applica o **OK**. L'opzione **Selezione automatica** è attiva per impostazione predefinita, e ti aiuta a scegliere un tema di colori più appropriato in base allo *schema di colori di KDE Plasma* che hai usato nelle applicazioni di modifica del testo. Di solito sceglie 'Brezza chiaro' o 'Brezza scuro', a seconda che lo schema sia rispettivamente chiaro o scuro.

SUGGERIMENTO

Puoi sistemare lo schema di colori globale di KDE nel modulo [Colori delle Impostazioni di sistema](#). In alcune applicazioni, tipo Kate o KDevelop, puoi anche modificarlo individualmente dal menu **Impostazioni** → **Schema di colori**.

6.3.5.1 Creare un nuovo tema

Per creare un nuovo tema è necessario prima copiarne uno esistente. Selezionane uno che vuoi usare come base, ad esempio 'Brezza chiaro' o 'Brezza scuro', e fai clic su **Copia**. Scrivi quindi il nome del nuovo tema.

Se vuoi modificare un tema integrato o a sola lettura, è necessario prima copiarlo con un nome diverso.

6.3.5.2 Importare o esportare i file di tema JSON

Puoi esportare un tema selezionato (inclusi quelli incorporati) in un [file JSON](#) con l'estensione **.theme** per mezzo del pulsante **Esporta**: si aprirà una finestra per salvare il file. Per aggiungere un tema di colori da un [file JSON](#), premi semplicemente il pulsante **Importa** e seleziona il file **.theme** dalla finestra.

SUGGERIMENTO

- Come [sopra menzionato](#), i file dei temi personalizzati dall'utente vengono salvati nella cartella `org.kde.syntax-highlighting/themes/`: quando copi o crei un tema, esso apparirà automaticamente lì. Inoltre l'importazione o l'aggiunta di un tema equivale alla copia di un file **.theme** esterno in questa cartella. KSyntaxHighlighting preleva automaticamente i file dei temi di colori da questa cartella.
- Se vuoi pubblicare un tema creato da te è essenziale controllare l'oggetto dei [metadati](#) del [file JSON](#), aggiungendo la rispettiva licenza e controllando il numero della revisione.

6.3.5.3 Modificare i temi di colori

6.3.5.3.1 Colori

Qui vengono regolati i colori dell'area di modifica del testo. È spiegato nei dettagli in Sezione [6.3.4.1](#).

6.3.5.3.2 Stili di testo predefiniti

Gli stili di testo predefiniti sono ereditati dagli stili di evidenziazione del testo, permettendo all'editor di presentare il testo in modo coerente; ad esempio, il testo dei commenti è uguale in quasi tutti i formati di testo che KSyntaxHighlighting può evidenziare.

Il nome della lista di stili usa lo stile configurato per tale elemento, fornendo un'anteprima quando si configura lo stile.

Ogni stile permette di selezionare degli attributi comuni oltre ai colori di sfondo e di primo piano. Per rimuovere l'impostazione del colore di sfondo, fai clic con il tasto destro del mouse per usare il menu contestuale.

Gli attributi di quest'area sono descritti in dettaglio in Sezione [6.3.4.2](#).

6.3.5.3.3 Stili di testo evidenziato

Qui si possono modificare gli stili del testo usati per una specifica definizione di evidenziazione. L'editor preseleziona l'evidenziazione usata per il documento attuale. Per modificare un'evidenziazione diversa, selezionarne una dalla casella combinata **Evidenziazione** sopra l'elenco degli stili.

Il nome della lista di stili usa lo stile configurato per tale elemento, fornendo un'anteprima quando si configura lo stile.

Ogni stile permette di selezionare attributi comuni oltre ai colori di sfondo e di primo piano. Per reimpostare un colore di sfondo, fai clic con il tasto destro del mouse per usare il menu contestuale. Puoi inoltre vedere se uno stile è uguale a quello predefinito dell'elemento, e renderlo tale se non lo è.

Noterai che molte evidenziazioni ne contengono altre, che sono rappresentate dai gruppi nell'elenco di stili. Ad esempio, la maggior parte delle evidenziazioni importano quella per Alert, e molte altre quella per Doxygen. La modifica dei colori di questi gruppi ha effetto sugli stili solo quando è usata nel formato di evidenziazione modificato.

6.3.6 Suggerimenti e trucchi

6.3.6.1 Contrasto dei colori del testo

Un aspetto importante quando si lavora coi temi di colori è quello di scegliere un contrasto del testo che renda più semplice la lettura, specialmente in combinazione con lo sfondo.

L'applicazione **Kontrast** è un controllore del contrasto: ti dice se le combinazioni di colore del testo e dello sfondo sono leggibili e accessibili, quindi è un eccellente strumento che ti aiuta a creare i temi di colori.

Puoi scaricare **Kontrast** dal [sito web delle applicazioni di KDE](#) o dal [pacchetto Flatpak su Flathub](#) (solo in GNU/Linux).

L'applicazione **Contrast** di GNOME è simile. Puoi scaricare [il pacchetto Flatpak su Flathub](#) (solo in GNU/Linux).

Uno script python nel [deposito di evidenziazione della sintassi](#) ti permette di visualizzare tutti i colori del tema, oltre al contrasto con i vari sfondi configurati.

6.3.6.2 Suggerimenti di consistenza per l'evidenziazione della sintassi

KSyntaxHighlighting include [più di 300 definizioni di evidenziazione della sintassi](#), quindi dovresti assicurarti che il tuo nuovo tema vada bene con tutte le definizioni di evidenziazione della sintassi. I temi di colori integrati hanno le similarità seguenti, che è raccomandabile (ma non obbligatorio) seguire per ottenere una visualizzazione corretta di tutte le definizioni di evidenziazione della sintassi:

- Usa il grassetto per le 'parole chiave' e per lo [stili del testo](#) del 'flusso di controllo'.
- Non usare un colore di sfondo in nessuno degli [stili di testo](#), tranne che in 'Alert' e in 'RegionMarker'.

La maggior parte degli evidenziatori di sintassi serve per dare un bell'aspetto nei temi predefiniti, 'Brezza chiaro' e 'Brezza scuro', quindi un altro modo per mantenere la consistenza è quello di usare dei colori simili negli [stili di testo](#), ad esempio il *verde* per il 'preprocessore' e 'altri', il *blu* per i 'tipi di dato' e per gli 'attributi', oppure il *viola* per le 'funzioni'.

Nota che queste raccomandazioni non sono obbligatorie nella creazione e nella pubblicazione di un tema.

6.4 Scripting con JavaScript

Il componente editor di KatePart è facilmente estendibile attraverso degli script. Il linguaggio di script è ECMAScript (comunemente noto come JavaScript). KatePart supporta due tipi di script: script di rientro e a riga di comando. Una funzionalità simile per comandare gli script da riga di comando viene anche fornita dall'[estensione dei frammenti](#).

6.4.1 Script di rientro

Gli script di rientro, noti anche come rientratori, fanno rientrare automaticamente il codice sorgente mentre si scrive. Per esempio, spesso il livello di rientro aumenta dopo aver premuto Invio.

Le sezioni seguenti descrivono passo-passo come creare la struttura di un semplice rientratore. Come primo passo crea un nuovo file *.js chiamato ad es. `javascript.js` nella cartella home locale `$XDG_DATA_HOME /katepart5/script/indentation`. Al suo interno la variabile di ambiente `XDG_DATA_HOME` viene tipicamente espansa sia in `~/.local` che in `~/.local/share`.

In Windows® questi file si trovano in `%USERPROFILE%\AppData\Local\katepart5\script\indentation`. `%USERPROFILE%` viene generalmente espanso in `C:\\Utenti\\utente`.

6.4.1.1 L'intestazione dello script di rientro

L'intestazione del file `javascript.js` è incorporata come JSON all'inizio del documento, e ha la forma seguente:

```
var katescript = {
  "name": "JavaScript",
  "author": "Pippo <pippo@topolinia.it>",
  "license": "BSD License",
  "revision": 1,
  "kate-version": "5.1",
  "required-syntax-style": "javascript",
  "indent-languages": ["javascript"],
  "priority": 0,
}; // kate-script-header, deve essere all'inizio della riga senza commenti
```

Ogni voce viene ora spiegata in dettaglio:

- `name` [obbligatorio]: questo è il nome del rientratore che appare nel menu **Strumenti** → **Rientro** e nella finestra di configurazione.
- `author` [facoltativo]: il nome dell'autore e informazioni per contattarlo.
- `license` [facoltativo]: forma breve della licenza, come Licenza BSD o LGPLv3.
- `revision` [obbligatorio]: la revisione dello script. Questo numero va aumentato ad ogni sua modifica.

- `kate-version` [obbligatorio]: versione minima di KatePart richiesta.
- `required-syntax-style` [facoltativo]: lo stile della sintassi richiesto, se soddisfa lo stile specificato nei file di evidenziazione della sintassi. È importante per i rientratori che richiedono informazioni di evidenziazione specifiche nel documento. Se uno stile di sintassi richiesto viene specificato, il rientratore è disponibile solo quando è attivo l'evidenziatore appropriato. Ciò impedisce il 'comportamento non definito' causato dall'uso del rientratore senza lo schema di evidenziazione atteso. Per esempio, il rientratore di Ruby lo usa nei file `ruby.js` e `ruby.xml`.
- `indent-languages` [facoltativo]: elenco JSON di stili di sintassi che il rientratore può far rientrare correttamente, ad es.: `[~c++, ~java]`.
- `priority` [facoltativo]: se ci sono più rientratori che si adattano a un certo file evidenziato la priorità decide quale viene scelto come predefinito.

6.4.1.2 Il codice sorgente del rientratore

Dopo aver specificato l'intestazione, questa sezione spiega come funziona lo script di rientro vero e proprio. La struttura fondamentale ha questo aspetto:

```
// librerie katepart js necessarie, ad es. range.js se usi Range
require ("range.js");

triggerCharacters = "{}/:;";
function indent(riga, larghezza_rientro, carattere)
{
    // richiamato a ogni ritorno a capo (carattere == '\n') e per tutti i ←
    // caratteri
    // specificati nella in variabile globale triggerCharacters. Quando si ←
    // chiama
    // StrumentiRientro formato
    // la variabile carattere è vuota, cioè caratteri == ''.
    //
    // Vedi anche: API di scripting
    return -2;
}
```

La funzione `indent()` ha tre argomenti:

- `riga`: la riga da far rientrare;
- `larghezza_rientro`: la larghezza del rientro espressa come numero di spazi;
- `carattere`: un carattere di ritorno a capo (`carattere == '\n'`), un carattere di attivazione specificato in `triggerCharacters`, o vuota se l'utente ha attivato l'azione **Strumenti** → **Rientro formato**.

Il valore restituito dalla funzione `indent()` specifica come far rientrare la riga. Se il valore è un semplice numero intero, viene interpretato come segue:

- Valore restituito -2: non fare nulla;
- Valore restituito -1: mantieni il rientro (cerca una riga non vuota precedente)
- Valore restituito 0: I numeri ≥ 0 specificano il rientro in spazi

In alternativa, si può restituire un array di due elementi:

- `return [indent, align];`

In questo caso, il primo elemento è il rientro, con lo stesso significato di cui sopra. Il secondo elemento, invece, è un valore assoluto che rappresenta una colonna di 'allineamento'. Se questo valore è maggiore del valore di rientro, la differenza rappresenta un numero di spazi da aggiungere dopo il rientro della prima variabile. Altrimenti, il secondo numero viene ignorato. Usare sia tabulatori che spazi per il rientro viene spesso indicato come 'modalità mista'.

Considera il seguente esempio: supponiamo di usare le tabulazioni per il rientro, e la loro ampiezza è impostata a 4. Qui, `<tab>` indica una tabulazione e «.» uno spazio:

```
1: <tab><tab>pippo("ciao",
2: <tab><tab>....."mondo");
```

Quando si fa rientrare la seconda riga, la funzione `indent()` restituisce [8, 15]. Perciò, si inseriscono due tabulazioni per far rientrare fino alla colonna 8, e vengono aggiunti sette spazi per allineare il secondo argomento al primo, in modo che rimanga allineato se il file viene visualizzato con un'ampiezza di tabulazione diversa.

Un'installazione predefinita di KDE include KatePart con diversi rientratori. Il codice JavaScript corrispondente si può trovare in `$XDG_DATA_DIRS /katepart5/script/indentation`.

In Windows® questi file si trovano in `%USERPROFILE%\AppData\Local\katepart5\script\indentation`. `%USERPROFILE%` viene generalmente espanso in `C:\\Utenti\\utente`.

Lo sviluppo di un rientratore richiede il ricaricamento gli script, per testare le modifiche. Invece di riavviare l'applicazione puoi però passare alla riga di comando, e digitare il comando **reload-scripts**.

Se sviluppi degli script utili, per piacere considera la possibilità di contribuire al progetto KatePart creando una richiesta di integrazione nel [nostro progetto GitLab](#).

6.4.2 Script da riga di comando

NOTA

Una funzionalità simile per comandare gli script da riga di comando viene anche fornita dall'[estensione dei frammenti](#). Questa può essere un punto di partenza più semplice, specialmente per i piccoli script personalizzati.

Essendo difficile soddisfare le necessità di tutti, KatePart supporta dei piccoli strumenti di supporto per manipolare velocemente il testo attraverso la [riga di comando integrata](#). Per esempio, il comando **sort** è implementato come uno script. Questa sezione spiega come creare file `*.js` per estendere KatePart con script di supporto a piacere.

Gli script da riga di comando si trovano nella stessa cartella degli script di rientro. Come primo passo, crea un nuovo file `*.js` chiamato `myutils.js` nella home locale `$XDG_DATA_HOME /katepart5/script/commands`. Al suo interno la variabile di ambiente `XDG_DATA_HOME` viene tipicamente espansa sia in `~/.local` che in `~/.local/share`.

In Windows® questi file si trovano in `%USERPROFILE%\AppData\Local\katepart5\script\commands`. `%USERPROFILE%` viene generalmente espanso in `C:\\Utenti\\utente`.

6.4.2.1 L'intestazione dello script per la riga di comando

L'intestazione di ciascuno script da riga di comando è incorporata in JSON all'inizio dello script come segue:

```
var katescript = {
  "author": "Example Name <example.name@some.address.org>",
  "license": "LGPLv2+",
  "revision": 1,
  "kate-version": "5.1",
  "functions": ["sort", "moveLinesDown"],
  "actions": [
    {
      "function": "sort",
      "name": "Ordina il testo selezionato",
      "category": "Editing",
      "interactive": "false"
    },
    {
      "function": "moveLinesDown",
      "name": "Sposta le righe in basso",
      "category": "Editing",
      "shortcut": "Ctrl+Shift+Down",
      "interactive": "false"
    }
  ]
}; // l'intestazione dello script di kate deve essere all'inizio del file ←
    senza commento
```

Ciascuna voce è spiegata ora in dettaglio:

- **author** [facoltativo]: il nome dell'autore e informazioni per contattarlo.
- **license** [facoltativo]: forma breve della licenza, come Licenza BSD o LGPLv2.
- **revision** [obbligatorio]: la revisione dello script. Questo numero va aumentato ad ogni sua modifica.
- **kate-version** [obbligatorio]: versione minima di KatePart richiesta.
- **functions** [obbligatorio]: vettore JSON di comandi nello script.
- **actions** [opzionale]: un vettore JSON di oggetti JSON che definiscono le azioni che appaiono nel menu dell'applicazione. Informazioni dettagliate sono fornite nella sezione [Stabilire le scorciatoie](#).

Dal momento che il valore di **functions** è un vettore JSON, un singolo script può contenere un numero arbitrario di comandi per la riga di comando. Ciascuna funzione è disponibile attraverso il menu [riga di comando integrata](#) di KatePart.

6.4.2.2 Il codice sorgente dello script

Tutte le funzioni specificate nell'intestazione devono essere implementate nello script. Lo script nell'esempio qui sopra deve implementare le due funzioni **sort** e **moveLinesDown**. Tutte le funzioni hanno la sintassi seguente:

```
// librerie katepart js necessarie, ad es. range.js se usi Range
require ("range.js");

function <nome>(argomento_1, argomento_2, ...)
{
  // ... implementazione, vedi anche: API per gli script
}
```


Gli argomenti nella riga di comando vengono passati alla funzione come *argomento_1*, *argomento_2*, ecc. Per poter documentare ogni comando, basta implementare la funzione `help` come segue:

```
function help(comando)
{
    if (comando == "sort") {
        return il8n("Ordina il testo selezionato.");
    } else if (comando == "...") {
        // ...
    }
}
```

Eseguire quindi **help sort** nella riga di comando chiamerà questa funzione ausiliaria con l'argomento *comando* impostato al comando dato, cioè *comando* == `"sort"`. KatePart presenterà quindi il testo risultante come documentazione per l'utente. Assicurati di [tradurre le stringhe](#).

Lo sviluppo di uno script da riga di comando richiede il ricaricamento gli script stesso, per testare le modifiche. Invece di riavviare l'applicazione puoi però passare alla riga di comando, e digitare il comando **reload-scripts**.

6.4.2.2.1 Stabilire le scorciatoie

Per rendere gli script accessibili nel menu dell'applicazione e per potergli assegnare delle scorciatoie, gli script devono fornire un'appropriata intestazione. Nell'esempio sottostante entrambe le funzioni `sort` e `moveLinesDown` appaiono nel menu grazie alla seguente parte dell'intestazione dello script:

```
var katescript = {
    ...
    "actions": [
        {
            "function": "sort",
            "name": "Ordina il testo selezionato",
            "icon": "",
            "category": "Editing",
            "interactive": "false"
        },
        {
            "function": "moveLinesDown",
            "name": "Sposta le righe in basso",
            "icon": "",
            "category": "Editing",
            "shortcut": "Ctrl+Shift+Down",
            "interactive": "false"
        }
    ]
};
```

I campi necessari sono i seguenti:

- **function** [obbligatorio]: la funzione che dovrebbe comparire nel menu **Strumenti** → **Script**.
- **name** [obbligatorio]: il testo che compare nel menu script.
- **icon** [facoltativo]: l'icona che appare di fianco al testo nel menu. Qui si possono usare tutti i nomi delle icone di KDE.
- **category** [facoltativo]: se si specifica una categoria, lo script compare in un sottomenu.
- **shortcut** [facoltativo]: la scorciatoia qui data è la predefinita, per esempio `Ctrl+Alt+t`. Vedi la [documentazione di Qt](#) per maggiori dettagli.

- `interactive` [facoltativo]: impostalo a `true` se allo script serve che l'utente faccia qualcosa.

Se sviluppi degli script utili, per piacere considera la possibilità di contribuire al progetto KatePart creando una richiesta di integrazione nel [nostro progetto GitLab](#).

6.4.3 API per gli script

L'API per script qui presentata è disponibile in tutti gli script, cioè gli script di rientro e i comandi da riga di comando. Le classi `Cursor` e `Range` sono fornite dalle librerie presenti in `$XDG_DATA_HOME /katepart5/libraries`. Se le vuoi usare nel tuo script, il che è necessario per usare alcune delle funzioni di `Document` o `View`, includi la libreria necessaria con:

```
// librerie katepart.js necessarie, ad es. range.js se usi Range
require ("range.js");
```

Per estendere l'API standard per gli script con funzioni e prototipi propri basta creare un nuovo file nella cartella di configurazione locale di KDE, `$XDG_DATA_HOME /katepart5/libraries`, e includerlo nel tuo script con:

```
require ("nome_del_mio_script.js");
```

In Windows® questi file si trovano in `%USERPROFILE%\AppData\Local\katepart5\libraries`. `%USERPROFILE%` viene generalmente espanso in `C:\Utenti\utente`.

Per estendere i prototipi preesistenti, come `Cursor` o `Range`, il modo raccomandato di procedere *non* è di modificare i file `*.js` globali. Cambia piuttosto il prototipo `Cursor` in JavaScript dopo aver incluso `cursor.js` nello script con `require`.

6.4.3.1 Cursori e intervalli

Essendo KatePart un editor di testo, tutta l'API per gli script si basa, ovunque sia possibile, su cursori e intervalli. Un cursore (`Cursor`) è una semplice tupla del tipo (`riga`, `colonna`) che rappresenta una posizione nel testo del documento. Un intervallo (`Range`) si estende sul testo a partire da una posizione di partenza a una finale del cursore. L'API viene spiegata in dettaglio nelle sezioni seguenti.

6.4.3.1.1 Il prototipo dei cursori

`Cursor()`;

Costruttore. Restituisce un cursore alla posizione (0, 0).

Esempio: `var cursore = new Cursor();`

`Cursor(int riga, int colonna)`;

Costruttore. Restituisce un cursore alla posizione (`riga`, `colonna`).

Esempio: `var cursore = new Cursor(3, 42);`

`Cursor(Cursor altro)`;

Costruttore di copia. Restituisce una copia dell'*altro* cursore.

Esempio: `var copia = new Cursor(altra);`

`Cursor Cursor.clone()`;

Restituisce un clone del cursore.

Esempio: `var clone = cursore.clone();`

Cursor.setPosition(int riga, int colonna);

Imposta la posizione del cursore a *riga* e *colonna*.

Da: KDE 4.11

bool Cursor.isValid();

Controlla se il cursore è valido. Non lo è se la riga o la colonna sono impostate a -1.

Esempio: `var valido = cursor.isValid();`

Cursor Cursor.invalid();

Restituisce un nuovo cursore non valido posizionato a (-1, -1).

Esempio: `var cursoreNonValido = cursor.invalid();`

int Cursor.compareTo(Cursor altro);

Confronta un cursore con un *altro*. Restituisce:

- -1, se il cursore è posizionato prima dell'*altro*,
- 0, se sono uguali, e
- +1, se il cursore è posizionato dopo l'*altro*.

bool Cursor.equals(Cursor altro);

Restituisce `true` se il cursore e l'*altro* sono uguali, altrimenti `false`.

String Cursor.toString();

Restituisce un cursore sotto forma di stringa nella forma 'Cursor(riga, colonna)'.

6.4.3.1.2 Il prototipo degli intervalli

Range();

Costruttore. Chiamare **new Range()** restituisce un intervallo tra (0, 0) e (0, 0).

Range(Cursor inizio, Cursor fine);

Costruttore. Chiamare `new Range(inizio, fine)` restituisce l'intervallo (*inizio*, *fine*).

Range(int riga_inizio, int colonna_inizio, int riga_fine, int colonna_fine);

Costruttore. Chiamare `new Range(riga_inizio, colonna_inizio, riga_fine, colonna_fine)` restituisce l'intervallo da (*riga_inizio*, *colonna_inizio*) a (*riga_fine*, *colonna_fine*).

Range(Range altro);

Costruttore di copia. Restituisce una copia dell'*altro* intervallo.

Range Range.clone();

Restituisce un clone dell'intervallo.

Esempio: `var clone = range.clone();`

bool Range.isEmpty();

Restituisce `true` se i cursori di inizio e fine sono uguali.

Esempio: `var empty = range.isEmpty();`

Da: KDE 4.11

bool Range.isValid();

Restituisce `true` se entrambi i cursori, iniziale e finale, sono validi, altrimenti `false`.

Esempio: `var valido = range.isValid();`

Range Range.invalid();

Restituisce l'intervallo da (-1, -1) a (-1, -1).

bool Range.contains(Cursor cursore);

Restituisce `true` se l'intervallo contiene la posizione del cursore, altrimenti `false`.

bool Range.contains(Range altro);

Restituisce `true` se l'intervallo contiene l'*altro*, altrimenti `false`.

bool Range.containsColumn(int colonna);

Restituisce `true` se la *colonna* è nell'intervallo semiaperto [inizio.colonna, fine.colonna), altrimenti `false`.

bool Range.containsLine(int riga);

Restituisce `true` se la *riga* è nell'intervallo semiaperto [inizio.riga, fine.riga), altrimenti `false`.

bool Range.overlaps(Range altro);

Restituisce `true` se l'intervallo e l'*altro* hanno una regione in comune, altrimenti `false`.

bool Range.overlapsLine(int riga);

Restituisce `true` se la *riga* è nell'intervallo [inizio.riga, fine.riga], altrimenti `false`.

bool Range.overlapsColumn(int colonna);

Restituisce `true` se la *colonna* è nell'intervallo [inizio.colonna, fine.colonna], altrimenti `false`.

bool Range.onSingleLine();

Restituisce `true` se l'intervallo comincia e finisce sulla stessa riga, cioè se *Range.start.line == Range.end.line*.

Da: KDE 4.9

bool Range.equals(Range altro);

Restituisce `true` se l'intervallo e l'*altro* sono uguali, altrimenti `false`.

String Range.toString();

Restituisce un intervallo sotto forma di stringa nella forma `'Range(Cursor(riga, colonna), Cursor(riga, colonna))'`.

6.4.3.2 Funzioni globali

Questa sezione elenca tutte le funzioni globali.

6.4.3.2.1 Leggere e includere file

String read(String file);

Cercherà nel *file* dato, relativamente alla cartella `katepart5/script/files`, e ne restituirà i contenuti in forma di stringa.

void require(String file);

Cercherà nel *file* dato, relativamente alla `katepart5/script/libraries`, e lo valuterà. `require` è già programmato per evitare inclusioni multiple dello stesso *file*.

Da: KDE 4.10

6.4.3.2.2 Debug

```
void debug(String testo);
```

Stampa il *testo* su `stdout` nella console che ha avviato l'applicazione.

6.4.3.2.3 Traduzione

Per supportare una localizzazione completa ci sono diverse funzioni per tradurre le stringhe negli script, cioè `i18n`, `i18nc`, `i18np` e `i18ncp`. Queste funzioni si comportano esattamente come le [funzioni di traduzione di KDE](#).

Le funzioni traducono le stringhe incluse con il sistema di traduzione di KDE nella lingua usata nell'applicazione. Le stringhe negli script sviluppati nel codice sorgente ufficiale di KatePart sono automaticamente estratte e traducibili: in altre parole gli sviluppatori di KatePart non devono preoccuparsi di estrarre e tradurre i messaggi. Va anche notato, però, che la traduzione funziona solo all'interno dell'infrastruttura di KDE: cioè nuove stringhe negli script di terze parti sviluppati al di fuori di KDE non sarebbero tradotte. Tuttavia prendi in considerazione l'idea di contribuire con i tuoi script a Kate, in modo che sia possibile una traduzione appropriata.

```
void i18n(String testo, argomento_1, ... );
```

Traduce *testo* nella lingua usata dall'applicazione. Gli argomenti *argomento_1* e seguenti sono facoltativi e vengono usati per sostituire i segnaposti `%1`, `%2`, ecc..

```
void i18nc(String contesto, String testo, argomento_1, ... );
```

Traduce *testo* nella lingua usata dall'applicazione. Inoltre, la stringa *contesto* viene resa visibile ai traduttori, per chiarire eventuali equivoci e produrre una traduzione migliore. Gli argomenti *argomento_1* e seguenti sono facoltativi, e vengono usati per sostituire i segnaposti `%1`, `%2` ecc..

```
void i18np(String singolare, String plurale, int numero, argomento_1, ... );
```

Traduce *singolare* o *plurale* nella lingua usata dall'applicazione, a seconda del *numero* dato. Gli argomenti *argomento_1* e seguenti sono facoltativi, e vengono usati per sostituire i segnaposti `%1`, `%2` ecc..

```
void i18ncp(String contesto, String singolare, String plurale, int numero, argomento_1, ... );
```

Traduce *singolare* o *plurale* nella lingua usata dall'applicazione, a seconda del *numero* o dato. Inoltre la stringa *contesto* viene resa visibile ai traduttori, per chiarire eventuali equivoci e produrre una traduzione migliore. Gli argomenti *argomento_1* e seguenti sono facoltativi, e vengono usati per sostituire i segnaposti `%1`, `%2` ecc..

6.4.3.3 API della vista

Ogni volta che uno script viene eseguito è presente una variabile globale, `'view'`, che rappresenta la vista attiva dell'editor. Segue un elenco di tutte le funzioni di vista disponibili.

```
void view.copy()
```

Taglia la selezione, se ce n'è una, altrimenti l'intera riga se è stata spuntata l'opzione [] **Copia o taglia la riga attuale se non c'è una selezione.**

Da: KDE Frameworks 5.79

```
void view.cut()
```

Taglia la selezione, se ce n'è una, altrimenti l'intera riga se è stata spuntata l'opzione [] **Copia o taglia la riga attuale se non c'è una selezione.**

Da: KDE Frameworks 5.79

void view.paste()

Incolla il contenuto degli appunti.

Da: KDE Frameworks 5.79

Cursor view.cursorPosition()

Restituisce la posizione attuale del cursore nella vista.

void view.setCursorPosition(int riga, int colonna); void

view.setCursorPosition(Cursor cursore);

Imposta la posizione attuale del cursore a (riga, colonna) o al *cursore* dato.

Cursor view.virtualCursorPosition();

Restituisce la posizione virtuale del cursore con ogni tabulazione che conta una quantità di spazi dipendente dall'attuale ampiezza di tabulazione.

void view.setVirtualCursorPosition(int riga, int colonna); void

view.setVirtualCursorPosition(Cursor cursore);

Imposta la posizione virtuale attuale del cursore a (riga, colonna) o al *cursore* dato.

String view.selectedText();

Restituisce il testo selezionato. Se non c'è del testo selezionato, la stringa restituita è vuota.

bool view.hasSelection();

Restituisce *true* se la vista contiene del testo selezionato, altrimenti *false*.

Range view.selection();

Restituisce l'intervallo di testo selezionato. L'intervallo di testo non è valido se non c'è testo selezionato.

void view.setSelection(Range intervallo);

Imposta il testo selezionato all'*intervallo* dato.

void view.removeSelectedText();

Rimuove il testo selezionato. Se la vista non ne ha non fa nulla.

void view.selectAll();

Seleziona tutto il testo del documento.

void view.clearSelection();

Pulisce la selezione di testo senza rimuoverlo.

void view.setBlockSelection(bool on);

Attiva o disattiva la modalità di selezione a blocchi.

bool view.blockSelection();

Restituisce *true* se la modalità di selezione a blocchi è attivata, altrimenti *false*.

void view.align(Range intervallo);

Fa rientrare nuovamente le righe che si trovano nell'*intervallo* in base alle impostazioni di rientro correnti.

void view.alignOn(Range intervallo, String modello = "");

Allinea le righe in *intervallo* alla colonna data dall'espressione regolare *modello*. Con un *modello* vuoto, per impostazione predefinita allineerà al primo carattere non di spazio. Se il modello ha una cattura, rientrerà

Esempi:

`view.alignOn(document.documentRange(), '-');` inserirà degli spazi prima del primo - di ogni riga per allinearle nella stessa colonna.

`view.alignOn(document.documentRange(), ':\s+(.)');` inserirà degli spazi prima del primo carattere non di spazio che ricorre dopo i due punti per allinearle nella stessa colonna.

object view.executeCommand(*String* comando, *String* argomenti, *intervallo*);

Esegue il [comando per la riga di comando](#) *comando* con gli argomenti opzionali *argomenti* e con l'opzionale *intervallo*. L'oggetto restituito ha la proprietà booleana *oggetto.ok*, che indica che l'esecuzione del *comando* ha avuto successo. In caso di errori, la stringa *oggetto.status* contiene un messaggio di errore.

Da: KDE Frameworks 5.50

Range view.searchText(*Range* intervallo, *String* modello, *bool* backwards = *false*);

Cerca la prima occorrenza di *modello* in *intervallo* e restituisce l'intervallo corrispondente. La ricerca viene eseguita all'indietro se il parametro booleano *backwards* è impostato a *true*.

L'intervallo restituito non è valido (vedi *Range.isValid()*) se *modello* non viene trovato in *intervallo*.

Da: KDE Frameworks 5.97

6.4.3.4 API del documento

Ogni volta che uno script viene eseguito è presente una variabile globale, 'document', che rappresenta il documento attivo. Segue un elenco di tutte le funzioni del documento disponibili.

String document.fileName();

Restituisce il nome del file del documento o una stringa vuota per i documenti non salvati.

String document.url();

Restituisce l'URL completo del documento, o una stringa vuota per i documenti non salvati.

String document.mimeType();

Restituisce il tipo di file MIME del documento, o il tipo MIME del file *application/octet-stream* se non se ne può trovare uno appropriato.

String document.encoding();

Restituisce la codifica attualmente usata per salvare il file.

String document.highlightingMode();

Restituisce la modalità di evidenziazione globale usata per tutto il documento.

String document.highlightingModeAt(*Cursor* posizione);

Restituisce la modalità di evidenziazione usata alla *posizione* nel documento.

Array document.embeddedHighlightingModes();

Restituisce un array di modalità di evidenziazione incorporate in questo documento.

bool document.isModified();

Restituisce *true* se il documento ha modifiche non salvate, altrimenti *false*.

String document.text();

Restituisce tutto il contenuto del documento in una sola stringa di testo. I ritorni a capo sono indicati con il carattere '\n'.

String document.text(*int* da_riga, *int* da_colonna, *int* a_riga, *int* a_colonna); String document.text(*Cursor* da, *Cursor* a); String document.text(*Range* intervallo);

Restituisce il testo nell'intervallo dato. Per migliorare la leggibilità del codice si raccomanda di usare le versioni con cursori o intervalli.

String document.line(int riga);

Restituisce la riga di testo richiesta come stringa. La stringa è vuota se la riga richiesta è oltre i limiti.

**String document.wordAt(int riga, int colonna); String
document.wordAt(Cursor cursore);**

Restituisce la parola alla posizione del cursore data.

**Range document.wordRangeAt(int riga, int colonna); Range
document.wordRangeAt(Cursor cursore);**

Restituisce l'intervallo di una parola alla posizione del cursore data. L'intervallo restituito non è valido (vedi Range.isValid()) se la posizione è oltre la fine di una riga. Se non c'è una parola alla fine del cursore viene restituito un intervallo vuoto.

Da: KDE 4.9

**String document.charAt(int riga, int colonna); String
document.charAt(Cursor cursore);**

Restituisce il carattere alla posizione del cursore data.

String document.firstChar(int riga);

Restituisce il primo carattere nella *riga* data che non sia uno spazio. Il primo carattere è alla colonna 0. Se la riga è vuota o contiene solo spazi la stringa restituita è vuota.

String document.lastChar(int riga);

Restituisce l'ultimo carattere nella *riga* data che non sia uno spazio. Se la riga è vuota o contiene solo spazi la stringa restituita è vuota.

bool document.isSpace(int riga, int colonna); bool document.isSpace(Cursor cursore);

Restituisce `true` se il carattere alla posizione del cursore data è uno spazio, altrimenti `false`.

**bool document.matchesAt(int riga, int colonna, String testo); bool
document.matchesAt(Cursor cursore, String testo);**

Restituisce `true` se il *testo* corrisponde a quello presente alla posizione del cursore, altrimenti `false`.

bool document.startsWith(int riga, String testo, bool salta_spazi);

Restituisce `true` se la riga comincia con il *testo*, altrimenti `false`. L'argomento *salta_spazi* decide se gli spazi iniziali vanno ignorati.

bool document.endsWith(int riga, String testo, bool salta_spazi);

Restituisce `true` se la riga finisce con il *testo*, altrimenti `false`. L'argomento *salta_spazi* decide se gli spazi finali vanno ignorati.

bool document.setText(String testo);

Imposta tutto il testo del documento.

bool document.clear();

Rimuove tutto il testo del documento.

bool document.truncate(int riga, int colonna); bool document.truncate(Cursor cursore);

Tronca la *riga* alla *colonna*. Restituisce `true` se funziona, o `false` se la *riga* non fa parte dell'intervallo del documento.

**bool document.insertText(int riga, int colonna, String testo); bool
document.insertText(Cursor cursore, String testo);**

Inserisce il *testo* alla posizione del cursore data. Restituisce `true` se funziona, o `false` se il documento è in modalità a sola lettura.


```
bool document.removeText(int da_riga, int da_colonna, int a_riga, int a_colonna); bool document.removeText(Cursor da, Cursor a); bool document.removeText(Range intervallo);
```

Rimuove il testo nell'intervallo dato. Restituisce `true` se funziona, o `false` se il documento è in modalità di sola lettura.

```
bool document.insertLine(int riga, String testo);
```

Inserisce il testo nella riga data. Restituisce `true` se funziona, o `false` se il documento è in modalità di sola lettura oppure la riga non è nell'intervallo del documento.

```
bool document.removeLine(int riga);
```

Rimuove la riga di testo data. Restituisce `true` se funziona, o `false` se il documento è in modalità di sola lettura oppure la riga non è nell'intervallo del documento.

```
bool document.wrapLine(int riga, int colonna); bool document.wrapLine(Cursor cursore);
```

Manda a capo la riga alla posizione del cursore data. Restituisce `true` se funziona, altrimenti `false`, ad es. se la riga < 0.

Da: KDE 4.9

```
void document.joinLines(int riga_inizio, int riga_fine);
```

Unisce le righe da *riga_inizio* a *riga_fine*. Due righe di testo consecutive sono sempre separate da un solo spazio.

```
int document.lines();
```

Restituisce il numero di righe nel documento.

```
bool document.isLineModified(int riga);
```

Restituisce `true` se la *riga* attuale contiene dati non salvati.

Da: KDE 5.0

```
bool document.isLineSaved(int riga);
```

Restituisce `true` se la *riga* è cambiata ma il documento è stato salvato, e quindi la riga corrente non contiene dati non salvati.

Da: KDE 5.0

```
bool document.isLineTouched(int riga);
```

Restituisce `true` se la *riga* contiene dati non salvati o è stata cambiata in precedenza.

Da: KDE 5.0

```
bool document.findTouchedLine(int startLine, bool giù);
```

Cerca la successiva riga modificata, a partire da *riga*. La ricerca è eseguita verso l'alto o verso il basso, a seconda della direzione specificata in *giù*.

Da: KDE 5.0

```
int document.length();
```

Restituisce il numero di caratteri nel documento.

```
int document.lineLength(int riga);
```

Restituisce la lunghezza della *riga*.

```
void document.editBegin();
```

Avvia un gruppo di modifica per un raggruppamento di azioni annullabili. Assicurati di chiamare sempre `editEnd()` con la stessa frequenza di `editBegin()`. Chiamando `editBegin` in viene usato internamente un contatore di riferimenti, cioè, questa chiamata può essere annidata.

```
void document.editEnd();
```

Chiude un gruppo di modifica. L'ultima chiamata di `editEnd()` (cioè quella corrispondente alla prima chiamata a `editBegin()`) conclude il passo di modifica.

int document.firstColumn(int riga);
Restituisce la prima colonna non di spazi nella *riga*. Se nella *riga* ci sono solo spazi, viene restituito -1.

int document.lastColumn(int riga);
Restituisce l'ultima colonna non di spazi nella *riga*. Se nella *riga* ci sono solo spazi, viene restituito -1.

int document.prevNonSpaceColumn(int riga, int colonna); int document.prevNonSpaceColumn(Cursor cursore);
Restituisce la colonna con caratteri non di spaziatura che comincia alla posizione del cursore data, cercando indietro.

int document.nextNonSpaceColumn(int riga, int colonna); int document.nextNonSpaceColumn(Cursor cursore);
Restituisce la colonna con caratteri non di spaziatura che comincia alla posizione del cursore data, cercando in avanti.

int document.prevNonEmptyLine(int riga);
Restituisce la prossima riga non vuota con caratteri non di spaziatura, cercando indietro.

int document.nextNonEmptyLine(int riga);
Restituisce la prossima riga non vuota con caratteri non di spaziatura, cercando in avanti.

bool document.isInWord(String carattere, int attributo);
Restituisce true se il *carattere* con l'*attributo* può far parte di una parola, altrimenti false.

bool document.canBreakAt(String carattere, int attributo);
Restituisce true se il *carattere* con l'*attributo* può essere mandato a capo, altrimenti false.

bool document.canComment(int attributo_inizio, int attributo_fine);
Restituisce true se un intervallo che inizia e finisce con gli attributi dati può essere fatto diventare un commento, altrimenti false.

String document.commentMarker(int attributo);
Restituisce l'indicatore di commento per i commenti di una sola riga per un *attributo*.

String document.commentStart(int attributo);
Restituisce l'indicatore di commento per l'inizio di commenti multi-riga per un *attributo*.

String document.commentEnd(int attributo);
Restituisce l'indicatore di commento per la fine di commenti multi-riga per un *attributo*.

Range document.documentRange();
Restituisce un intervallo che comprende tutto il documento.

Cursor document.End();
Restituisce un cursore posizionato nell'ultima colonna dell'ultima riga del documento.

bool isValidTextPosition(int riga, int colonna); bool isValidTextPosition(Cursor cursore);
Restituisce true se la posizione del cursore ricade in una posizione di testo valida, cioè se è localizzata all'inizio, nel mezzo o alla fine di una riga valida. Una posizione di testo non è valida se si trova in un surrogato Unicode.

Da: KDE 5.0

int document.attribute(int riga, int colonna); int document.attribute(Cursor cursore);
Restituisce l'attributo alla posizione del cursore data.

```
bool document.isAttribute(int riga, int colonna, int attributo); bool
document.isAttribute(Cursor cursore, int attributo);
```

Restituisce `true` se l'attributo alla posizione del cursore data è uguale a `attributo`, altrimenti `false`.

```
String document.attributeName(int riga, int colonna); String
document.attributeName(Cursor cursore);
```

Restituisce il nome dell'attributo in testo leggibile. È uguale al nome `itemData` nei file di evidenziazione della sintassi.

```
bool document.isAttributeName(int riga, int colonna, String nome); bool
document.isAttributeName(Cursor cursore, String nome);
```

Restituisce `true` se il nome dell'attributo a una certa posizione del cursore corrisponde al `nome`, altrimenti `false`.

```
String document.variable(String chiave);
```

Restituisce il valore della variabile del documento `chiave`. Se la variabile non esiste il valore restituito è una stringa vuota.

```
void document.setVariable(String chiave, String valore);
```

Imposta il valore della variabile del documento richiesta `chiave`.
Vedi anche: [le variabili nei documenti di Kate](#).
Da: KDE 4.8

```
int document.firstVirtualColumn(int riga);
```

Restituisce la colonna virtuale del primo carattere non di spaziatura nella riga indicata, o -1 se la riga è vuota oppure contiene solo caratteri di spaziatura.

```
int document.lastVirtualColumn(int riga);
```

Restituisce la colonna virtuale dell'ultimo carattere non di spaziatura nella riga indicata, o -1 se la riga è vuota oppure contiene solo caratteri di spaziatura.

```
int document.toVirtualColumn(int riga, int colonna);
int document.toVirtualColumn(Cursor cursore); Cursor
document.toVirtualCursor(Cursor cursore);
```

Converte la posizione 'reale' del cursore in una virtuale, restituendo un oggetto `int` o `Cursor`.

```
int document.fromVirtualColumn(int riga, int colonna_virtuale);
int document.fromVirtualColumn(Cursor cursore_virtuale); Cursor
document.fromVirtualCursor(Cursor cursore_virtuale);
```

Converte la posizione virtuale data del cursore in una 'reale', restituendo un oggetto `int` o `Cursor`.

```
Cursor document.anchor(int riga, int colonna, Char carattere); Cursor
document.anchor(Cursor cursore, Char carattere);
```

Cerca indietro il carattere partendo dal cursore dato. Per esempio, se si passa «(» come carattere, la funzione restituirà la posizione dell'apertura «(». Questo conteggio dei riferimenti, cioè altri «(...)», vengono ignorati.

```
Cursor document.rfind(int riga, int colonna, String testo, int attributo =
-1); Cursor document.rfind(Cursor cursore, String testo, int attributo =
-1);
```

Cerca all'indietro il testo con l'`attributo` appropriato. L'`attributo` viene ignorato se è impostato a -1. Il cursore restituito non è valido se non si trova il testo.

```
int document.defStyleNum(int riga, int colonna); int
document.defStyleNum(Cursor cursore);
```

Restituisce lo stile predefinito usato alla posizione data del cursore.

```
bool document.isCode(int riga, int colonna); bool document.isCode(Cursor cursore);
```

Restituisce `true` se l'attributo alla posizione data del cursore non è uguale a tutti i seguenti stili: `dsComment`, `dsString`, `dsRegionMarker`, `dsChar`, `dsOthers`.

```
bool document.isComment(int riga, int colonna); bool document.isComment(Cursor cursore);
```

Restituisce `true` se l'attributo del carattere alla posizione del cursore data è `dsComment`, altrimenti `false`.

```
bool document.isString(int riga, int colonna); bool document.isString(Cursor cursore);
```

Restituisce `true` se l'attributo del carattere alla posizione del cursore data è `dsString`, altrimenti `false`.

```
bool document.isRegionMarker(int riga, int colonna); bool document.isRegionMarker(Cursor cursore);
```

Restituisce `true` se l'attributo del carattere alla posizione del cursore data è `dsRegionMarker`, altrimenti `false`.

```
bool document.isChar(int riga, int colonna); bool document.isChar(Cursor cursore);
```

Restituisce `true` se l'attributo del carattere alla posizione del cursore data è `dsChar`, altrimenti `false`.

```
bool document.isOthers(int riga, int colonna); bool document.isOthers(Cursor cursore);
```

Restituisce `true` se l'attributo del carattere alla posizione del cursore data è `dsOthers`, altrimenti `false`.

```
void document.indent(Range intervallo, int modifica);
```

Fa rientrare tutte le righe nell'*intervallo* con la *modifica* delle schede o del numero delle volte degli spazi `tabSize`, a seconda delle preferenze dell'utente. Il parametro di *modifica* può essere negativo.

6.4.3.5 L'API dell'editor

Oltre alle API per documento e vista c'è un'API generale per editor, che fornisce delle funzioni per le funzionalità di script generali per editor.

```
String editor.clipboardText();
```

Restituisce il testo che c'è attualmente negli appunti globali.

Da: KDE Frameworks 5.50

```
String editor.clipboardHistory();
```

L'editor mantiene una cronologia degli appunti che contiene fino a 10 voci di appunti. Questa funzione restituisce tutte le voci che ci sono attualmente nella cronologia degli appunti.

Da: KDE Frameworks 5.50

```
void editor.setClipboardText(String testo);
```

Imposta il contenuto degli appunti a *testo*. il *testo* sarà aggiunto alla cronologia degli appunti.

Da: KDE Frameworks 5.50

Capitolo 7

Configura KatePart

Selezionando **Impostazioni** → **Configura Applicazione...** dal menu, si apre la finestra di dialogo **Configura**. Questa finestra di dialogo può essere utilizzata per modificare diverse impostazioni. Le impostazioni disponibili da cambiare variano a seconda della quale categoria scelta nella lista verticale sul lato sinistro della finestra. Per mezzo di tre pulsanti lungo la base della finestra l'utente può controllare il processo.

Puoi invocare il sistema di **Aiuto**, accettare le impostazioni correnti e chiudere la finestra di dialogo con il pulsante **OK** oppure, con **Annulla**, annullare il processo. Le categorie **Aspetto**, **Caratteri e colori**, **Modifica**, **Apri e salva** e **Estensioni** sono descritte sotto.

7.1 Configurazione del componente editor

Questo gruppo contiene tutte le pagine relative al componente editor di KatePart. La maggior parte delle impostazioni sono quelle predefinite, che possono venir sostituite [definendo un tipo di file](#), con le [variabili dei documenti](#) o cambiandole nel documento in una sessione di lavoro.

7.1.1 Aspetto

7.1.1.1 Generale

Carattere dell'editor

Qui puoi scegliere il tipo di carattere dell'editor di testo. È possibile scegliere qualsiasi carattere disponibile nel sistema ed impostare una dimensione predefinita. Un testo di esempio è mostrato nella parte inferiore della finestra, per valutare gli effetti delle tue scelte.

Per maggiori informazioni su come selezionare un carattere, vedi la [sezione Scegliere i caratteri della documentazione sui Fondamentali di KDE](#).

Mostra gli indicatori degli spazi

Mai

L'editor non mostrerà mai dei punti per indicare la presenza di spazi.

Alla fine di una riga

L'editor mostrerà dei punti per indicare la presenza di spazi finali in fondo alle righe.

Sempre

L'editor mostrerà sempre dei punti per indicare la presenza di spazi.

Dimensioni indicatore degli spazi

Usa il cursore per modificare la dimensione del marcatore dell'indicatore visibile.

Mostra indicatori delle schede

Se spuntato l'editor mostra un simbolo » per indicare la presenza di una tabulazione nel testo.

Verifica delle parentesi

Evidenzia intervallo tra parentesi selezionate

Se abilitato, l'intervallo tra le parentesi selezionate corrispondenti sarà evidenziato.

Mostra un'anteprima della parentesi aperta corrispondente

Se abilitato, l'editor mostra un suggerimento della parentesi aperta corrispondente.

Fai lampeggiare la parentesi corrispondente quando il cursore si sposta sull'altra parentesi della coppia

Se abilitato, un passaggio sopra le parentesi ({, [,], },(or)) farà lampeggiare rapidamente la parentesi corrispondente.

Mostra righe di rientro

Se spuntata, l'editor mostrerà delle righe verticali per favorire l'identificazione delle righe rientrate.

Conteggi

Mostra il conteggio delle parole

Visualizza il numero di parole e di caratteri nel documento e nella selezione corrente nella barra di stato. Questa opzione è disponibile anche nel menu contestuale della barra di stato.

Mostra il numero delle righe

Visualizza nella barra di stato il numero totale delle righe del documento. Questa opzione è disponibile anche nel menu contestuale della barra di stato.

Contrai la prima riga

Se abilitato, la prima riga è contratta, se possibile. Questo è utile se il file inizia con un commento, ad esempio un copyright.

A capo automatico dinamico

Se quest'opzione è spuntata, le righe proseguiranno a capo quando raggiungeranno il bordo della finestra sullo schermo.

Ritorno a capo dinamico al marcatore di ritorno a capo statico

Se abilitato, l'editor fa andare a capo dinamicamente nella [posizione dell'andata a capo dinamica](#).

Ignora i limiti delle parole per il ritorno a capo dinamico

Se selezionato, l'editor non terrà conto dei limiti delle parole facendo andare a capo le righe di testo.

Indicatori dinamici di ritorno a capo

Scegli quando mostrare gli indicatori di a capo automatico dinamico, **Spenti**, **Dopo i numeri di riga** o **Sempre attivi**.

Rientra righe a capo

Inoltre permette di impostare la massima larghezza dello schermo come percentuale dopo la quale le righe andate a capo dinamicamente non saranno più allineate verticalmente. Ad esempio, a 50% le righe il cui livello di rientro è più profondo del 50% della larghezza dello schermo non saranno allineate verticalmente alle righe seguenti che sono andate a capo automaticamente.

Moltiplicatore dell'altezza della linea

Questo valore sarà moltiplicato per l'altezza della linea predefinita del carattere. Un valore di 1,0 significa che sarà utilizzata l'altezza predefinita.

7.1.1.2 Bordi

Raggruppamento dei blocchi di codice

Mostra le frecce per contrarre i blocchi di codice

Se questa opzione è spuntata, la vista corrente mostrerà degli indicatori per il raggruppamento del codice, se il raggruppamento del codice è disponibile.

Mostra un'anteprima del codice raggruppato al passaggio del puntatore

Se spuntata, l'anteprima del testo ripiegato viene mostrata in una finestra a comparsa al passaggio sopra ad una regione ripiegata

Visibilità delle frecce di espansione

Cambia le frecce di espansione tra **Mostra al passaggio del puntatore** e **Mostra sempre**.

Lato sinistro

Mostra i marcatori

Se spuntata, sarà visualizzato sul lato sinistro un bordo per le icone. Il bordo per le icone mostra, ad esempio, i marcatori dei segnalibri.

Mostra numeri di riga

Se spuntata, sul lato sinistro saranno visualizzati i numeri di riga.

Evidenzia le righe modificate e non salvate

Se attivata, saranno visibili i segni di modifica delle righe. Per maggiori informazioni, vedi Sezione [3.9](#).

Barre di scorrimento

Mostra i marcatori

Se questa opzione è spuntata, la vista attuale mostrerà indicatori sulla barra di scorrimento verticale. Questi segni possono mostrare, ad esempio, i segnalibri.

Mostra un'anteprima passando sulla barra di scorrimento

Se questa opzione è spuntata e passi con il cursore del mouse sopra la barra di scorrimento, verrà visualizzata una piccola anteprima del testo di alcune righe intorno la posizione del cursore. Questo ti permette di passare rapidamente ad un'altra parte del documento.

Minimappa

Mostra la minimappa

Se questa opzione è selezionata, ogni nuova vista mostrerà una minimappa del documento sulla barra di scorrimento verticale.
Per maggiori informazioni sulla minimappa di scorrimento, vedi Sezione [3.10](#).

Larghezza della minimappa

Regola la larghezza della minimappa di scorrimento, definita in pixel.

Visibilità delle barre di scorrimento

Attiva le barre di scorrimento, disattiva o mostrale solo se sono necessarie. Fai clic con il tasto sinistro del mouse sul rettangolo blu per visualizzare l'intervallo del numero di righe del documento che sono visualizzate sullo schermo. Tieni il tasto sinistro del mouse premuto al di fuori del rettangolo blu per scorrere automaticamente il documento

Ordina il menu dei segnalibri

Per data di creazione

Ogni nuovo segnalibro verrà aggiunto in coda, indipendentemente da dove è collocato nel documento.

Per numero di riga

I segnalibri saranno ordinati a seconda dei numeri di riga a cui sono collocati.

7.1.2 Temi di colori

Questa sezione della finestra ti permette di configurare tutti i colori in qualsiasi tema tu abbia, di creare nuovi schemi o di eliminare quelli esistenti, oppure semplicemente di **Seguire lo schema di colori di sistema**. Ciascuno schema ha le impostazioni per i colori, per i tipi di carattere e per gli stili del testo normale o evidenziato.

KatePart preselezionerà per te il tema attualmente attivo. Se vuoi lavorare su uno schema diverso, inizia selezionandolo dalla casella combinata **Seleziona tema**. Con i pulsanti **Copia** ed **Elimina** puoi creare nuovi temi ed eliminare quelli esistenti.

È descritto nei dettagli in Sezione [6.3.5](#).

7.1.3 Modifica

7.1.3.1 Generale

A capo automatico

L'a capo automatico è una funzionalità che fa in modo che l'editor inizi automaticamente una nuova riga di testo spostando (a capo) il cursore all'inizio di quella nuova riga. KatePart comincerà automaticamente una nuova riga di testo quando la riga attuale raggiungerà la lunghezza specificata dall'opzione [Torna a capo dopo](#).

Ritorna a capo a una colonna prefissata

Attiva o disattiva l'a capo automatico statico.

Disegna una linea verticale alla colonna di ritorno a capo

Se questa opzione è spuntata, verrà disegnata una barra verticale sulla colonna di andata a capo definita in **Impostazioni** → **Configura editor...** nella scheda Modifica. Attenzione che l'indicatore di andata a capo sarà disegnato solo se utilizzi un carattere a spaziatura fissa.

Torna a capo dopo:

Se l'opzione [Ritorna a capo a una colonna prefissata](#) è selezionata, questa voce determina la lunghezza (in caratteri) dopo la quale l'editor inizierà automaticamente una nuova riga.

Modalità di inserimento predefinita

La modalità di inserimento selezionata sarà abilitata quando si aprirà una nuova vista. Puoi sempre attivare o disattivare la modalità di inserimento Vi per una vista particolare nel menu **Modifica**.

Parentesi

Se l'opzione **Chiudi automaticamente le parentesi quando si digita la parentesi aperta**, quando l'utente digita una parentesi sinistra ([, (, o {) KatePart inserirà automaticamente la parentesi destra (],), o }) alla destra del cursore.

Caratteri di delimitazione

È possibile selezionare i caratteri di delimitazione usando l'elenco a tendina corrispondente. Quando il testo è selezionato, la digitazione di uno di questi caratteri spezza il testo selezionato.

Copia e incolla

Sposta il testo selezionato con il trascinamento

Questa opzione abilita il trascinamento del testo selezionato nella finestra dell'editor.

Copia/taglia la riga attuale senza un testo selezionato

Se questa opzione è abilitata e la selezione è vuota, le operazioni di copia e taglia sono effettuate sulla riga in cui si trova il cursore.

Non spostare il cursore del testo quando si incolla con il mouse

Se questa opzione è abilitata e incolli del testo nella finestra dell'editor con un clic sul tasto centrale del mouse, KatePart non sposterà il cursore del testo alla posizione in cui hai fatto clic.

7.1.3.2 Navigazione del testo

Movimento del cursore di testo

Home e Fine intelligenti

Se è spuntato, la pressione del tasto Home farà saltare al cursore gli spazi bianchi e lo farà posizionare all'inizio del testo della riga.

PagSu e PagGiù spostano il cursore

Questa opzione cambia il comportamento del cursore quando l'utente preme il tasto **Pag↑** o **Pag↓**. Se non è selezionato, il cursore del testo manterrà la sua posizione relativamente al testo visibile in KatePart, quando del nuovo testo diventerà visibile come risultato dell'operazione. Così se il cursore è al centro del testo visibile, quando avviene l'operazione, rimarrà lì (eccetto quando si raggiunge l'inizio o la fine). Quando questa opzione è selezionata, il primo tasto premuto causerà lo spostamento del cursore alla cima o alla base del testo visibile non appena si visualizza una nuova pagina di testo.

Abilita il movimento del cursore sulla notazione delle maiuscole a cammello

Questa opzione modifica il comportamento del cursore quando l'utente preme la scorciatoia **Ctrl-Freccia sinistra** o **Ctrl-Freccia destra**. Se non è selezionata il cursore salta parole intere, mentre se lo è il cursore si interrompe sulle gobbe della notazione delle maiuscole a cammello.

Centrata automatica del cursore:

Imposta il numero di righe che, se possibile, devono rimanere visibili sopra e sotto il cursore.

Modalità di selezione del testo

Normale

Le selezioni vengono sovrascritte con il testo digitato e vanno perse quando si muove il cursore.

Persistente

Le selezioni rimangono anche dopo lo spostamento del cursore e l'inserimento di testo.

Permetti di scorrere oltre la fine del documento

Questa opzione ti permette di scorrere oltre la fine del documento. Può essere usato per centrare verticalmente la parte bassa del documento, o portarla nella parte alta della vista.

Il tasto Backspace rimuove la base del carattere con i suoi diacritici

Se selezionato, i caratteri composti sono rimossi insieme ai loro diacritici invece di rimuovere solo la base del carattere. Utile per le localizzazioni indiane.

Modificatore multi cursore

Questa opzione ti permette di impostare il modificatore che verrà usato per creare più cursori con un clic del tasto sinistro del mouse: devi premerlo e fare clic sul tasto sinistro del mouse per creare un cursore alla posizione desiderata. Vedi [Creazione di più cursori](#) per scoprire altre modalità per creare più cursori.

7.1.3.3 Rientro

Modalità di rientro predefinita:

Seleziona la modalità di rientro automatico da usare come predefinita. Si raccomanda caldamente di impostare **Nessuno** o **Normale** e di utilizzare le configurazioni specifiche per tipo di file per impostare altre modalità di rientro per formati di testo come codice C/C++ o XML.

Rientra con

Tabulazioni

Quando abilitato, l'editor inserirà caratteri di tabulazione quando premi il tasto **Tab** o usi il [rientro automatico](#).

Spazi

Quando questa opzione è abilitata, l'editor inserirà un numero appropriato di spazi a seconda della posizione nel testo e dell'impostazione **Ampiezza delle tabulazioni** alla pressione del tasto **Tab** o usi il [rientro automatico](#).

Tabulazioni e spazi

Quando abilitato, l'editor inserirà spazi come sopra descritto quando rientri o premi il tasto **Tab** all'inizio di una riga, ma inserirà tabulazioni quando il tasto **Tab** viene premuto all'interno o alla fine di una riga.

Ampiezza delle tabulazioni:

Configura il numero di spazi mostrati al posto di un carattere di tabulazione.

Larghezza del rientro:

La larghezza del rientro è il numero di spazi usati per fare rientrare una riga. Se configurato per rientrare con le tabulazioni, viene inserito un carattere di tabulazione se il rientro è divisibile per l'ampiezza delle tabulazioni.

Regole di rientro

Mantieni gli spazi di troppo

Se questa opzione non è spuntata, la modifica del livello di rientro allineerà le righe a multipli della larghezza specificata in **Larghezza di rientro**.

Regola il rientro del testo incollato dagli appunti

Se questa opzione è spuntata, il testo incollato dagli appunti verrà fatto rientrare. Puoi rimuovere il rientro con il comando **Annulla**.

Azioni di rientro

Il tasto Backspace nello spazio vuoto iniziale toglie il rientro

Se è spuntato, il tasto **Backspace** diminuisce il livello di rientro se il cursore si trova nello spazio iniziale di una riga.

Azione del tasto Tab (senza selezione)

Se vuoi che **Tab** allinei la riga attuale nel blocco di codice attuale come in Emacs, puoi rendere **Tab** una scorciatoia per l'azione **Rientro formato**.

Avanza sempre alla prossima tabulazione

Se è spuntato, il tasto **Tab** inserirà sempre spazio bianco in modo da raggiungere la prossima tabulazione. Se l'opzione **Inserisci spazi invece di tabulazioni** nella pagina **Modifica** è abilitata, vengono inseriti spazi; altrimenti, viene inserita una tabulazione.

Aumenta sempre il livello di rientro

Se è spuntato, il tasto **Tab** farà rientrare sempre la riga attuale del numero di posizioni specificato in **Larghezza di rientro**.

Aumenta il livello di rientro se in uno spazio vuoto iniziale

Se è spuntato, il tasto **Tab** farà rientrare la riga attuale o avanzerà alla prossima tabulazione. Se il punto di inserimento è sul primo carattere non di spaziatura della riga (o prima), o se c'è una selezione, la riga corrente verrà fatta rientrare del numero di posizioni specificato in **Larghezza di rientro**. Se il punto di inserimento è dopo il primo carattere non di spaziatura della linea e non c'è selezione, viene inserito dello spazio bianco in modo da raggiungere la prossima tabulazione: se l'opzione **Inserisci spazi invece di tabulazioni** nella scheda **Generale** della pagina **Modifica** è abilitata, vengono inseriti spazi; altrimenti, viene inserita una tabulazione.

7.1.3.4 Completamento delle parole

Generale

Abilita il completamento delle parole

Se è abilitato, si aprirà automaticamente un riquadro di completamento durante la digitazione, che visualizza un elenco di modi in cui è possibile completare il testo presente alla posizione del cursore.

Seleziona automaticamente la prima voce di completamento

Se abilitato, la prima voce di completamento automatico viene sempre preselezionata, così da poter essere inserita con **Invio**. Se non vuoi questo comportamento, ad es. se vuoi premere **Invio** solo per inserire una nuova riga, disabilita questo elemento.

Lunghezza minima delle parole da completare

Mentre si digita, il completamento automatico cerca nel documento le parole che iniziano con i caratteri già digitati. Questa opzione permette di configurare il numero minimo di caratteri che bisogna digitare prima che il completamento automatico si attivi e visualizzi le corrispondenze trovate.

Rimuovi coda al completamento

Rimuovi la coda di una parola precedente quando l'elemento di completamento viene scelto da un elenco.

Completamento delle parole

Se abilitato, il completamento automatico integrato usa le parole chiave definite dall'evidenziazione della sintassi.

7.1.3.5 Controllo ortografico

Queste opzioni di configurazione sono spiegate nella documentazione del modulo [Controllo ortografico](#) di Impostazioni di sistema .

7.1.3.6 Modalità di inserimento Vi

Generale

Lascia che i comandi Vi abbiano precedenza su quelli di Kate

Se selezionato, i comandi Vi avranno precedenza sui comandi interni di KatePart. Per esempio, **Ctrl+R** rifarà le modifiche, e sostituirà l'azione normale, che sarebbe mostrare la finestra di ricerca e sostituzione.

Visualizza i numeri di riga relativi

Se abilitato, la riga attuale diventa la riga 0. Le righe sopra e sotto incrementano il numero relativamente.

Mappatura dei tasti

La mappatura dei tasti è usata per cambiare il significato dei tasti mappati. Questo permette di spostare comandi su altri tasti o impostare dei tasti in modo da eseguire una serie di comandi.

Esempio:

F2 → **I--** Esc

Questo anteporrà un **I--** a una riga quando si preme **F2**.

7.1.4 Apri e salva

7.1.4.1 Generale

Formato dei file

Codifica

Questo definisce la codifica normale da usare per aprire e salvare i file, se non viene modificata nelle finestre di apertura e salvataggio o usando un'opzione dalla riga di comando.

Rilevamento della codifica

Selezionare un elemento dal menu a discesa, per disabilitare il rilevamento automatico o per usare **Universale** per abilitare il rilevamento automatico per tutte le codifiche. Dato che probabilmente rileverà solo utf-8/utf-16, è meglio selezionare una regione per poter indovinare meglio. Se né la codifica scelta come normale sopra, né quella specificata nella finestra di apertura e salvataggio, né quella specificata sulla riga di comando corrispondono ai contenuti del file, verrà usato questo rilevamento.

Codifica di riserva

Qui si definisce la codifica di riserva per provare ad aprire file se né la codifica sopra indicata come normale, né quella specificata nella finestra di apertura o salvataggio, né quella specificata sulla riga di comando corrispondono ai contenuti del file. Prima di usare questa, si proverà a determinare la codifica da usare cercando un indicatore di ordinamento dei byte all'inizio del file: se ne si trova uno, sarà scelta la codifica Unicode corretta; altrimenti, verrà avviato il rilevamento della codifica, se entrambe le codifiche di riserva saranno provate.

Fine riga

Sceglie la modalità di fine riga preferita per il documento attuale. Puoi scegliere tra UNIX®, DOS/Windows® oppure Macintosh.

Rilevamento automatico delle fini di riga

Spuntalo se vuoi che l'editor riconosca automaticamente il tipo di fine riga. Il primo tipo di fine riga incontrato verrà usato per tutto il file.

Abilita il marcatore dell'ordine dei byte (BOM)

Il marcatore dell'ordine dei byte è una sequenza speciale all'inizio dei documenti unicode. Permette ai programmi di aprire i documenti di testo con le impostazioni unicode corrette. Per maggiori informazioni, vedere [Byte Order Mark](#).

Limite di lunghezza delle righe

Sfortunatamente, a causa di mancanze dovute a Qt™, KatePart non funziona bene con righe estremamente lunghe. Per quel motivo, KatePart manderà a capo automaticamente le righe più lunghe del numero di caratteri qui specificato. Per disattivarlo, impostalo a 0.

Pulizia automatica al salvataggio

Rimuovi gli spazi finali durante la scrittura

L'editor eliminerà automaticamente gli spazi di troppo alla fine delle righe di testo durante il salvataggio del file. Puoi selezionare di non usare **Mai** questa funzionalità, o di usarla solo **Sulle righe modificate** dall'ultimo salvataggio del documento, o di rimuoverli incondizionatamente **In tutto il documento**.

Aggiungi un ritorno a capo alla fine del file al salvataggio

L'editor aggiungerà automaticamente un ritorno a capo alla fine del file se non ce n'è già uno al momento del salvataggio.

Abilita il salvataggio automatico (solo file locali)

Spuntalo se vuoi che l'editor salvi automaticamente i documenti mentre ci stai lavorando.

Salva automaticamente il documento quando il fuoco lascia l'editor

L'editor salverà automaticamente i documenti quando passi ad un elemento al di fuori di esso, ad es., il pannello del terminale di Kate.

Intervallo di salvataggio automatico

Qui puoi determinare l'intervallo per il salvataggio automatico in secondi: se è 0, il documento non verrà salvato automaticamente dopo tale intervallo.

7.1.4.2 Avanzate

Scrivi una copia di sicurezza al salvataggio di

La copia di sicurezza al salvataggio fa copiare a KatePart il file su disco (la sua versione precedentemente salvata) in <prefisso><nomefile><uffisso> prima di salvare le nuove modifiche. Una copia di sicurezza può aiutarti a recuperare il lavoro se qualcosa va storto nel salvataggio, oppure se in seguito vuoi recuperare la versione precedente al salvataggio. Il suffisso predefinito è ~, ed il prefisso è vuoto.

File locali

Spunta la casella se vuoi una copia di sicurezza dei file locali quando salvi.

File remoti

Spunta la casella se vuoi una copia di sicurezza dei file remoti quando salvi.

Prefisso per le copie di sicurezza

Inserisci il prefisso per i nomi delle copie di sicurezza.

Suffisso per le copie di sicurezza

Inserisci il suffisso per i nomi delle copie di sicurezza.

Modalità file di swap

KatePart può recuperare (la maggior parte) del lavoro non salvato in caso di arresto anomalo o di mancanza di corrente. Quando un documento viene modificato viene creato un file di swap (`.<nomefile>.kate-swp`): se l'utente non salva le modifiche e KatePart subisce un arresto anomalo, questo rimane invece sul disco. All'apertura di un file KatePart controlla se c'è un file di swap associato: in caso affermativo chiede all'utente se voglia o meno recuperare i dati persi. C'è anche la possibilità di vedere le differenze tra il file originale e quello recuperato. Il file di swap viene eliminato dopo ogni salvataggio, e alla normale uscita dal programma.

KatePart sincronizza i file di swap sul disco ogni 15 secondi, ma solo se sono cambiati dall'ultima sincronizzazione. L'utente può scegliere di disabilitare la sincronizzazione, spuntando la casella **Disabilita**, ma ciò può portare a una perdita di dati maggiore.

Quando il file di swap è abilitato, è possibile passare tra due modalità, chiamate **Abilitata, archivia nella cartella predefinita** e **Abilitata, archivia in una cartella personalizzata**.

Archivia i file di swap in

Per impostazione predefinita i file di swap vengono salvati nella stessa cartella del file. Se viene come modalità di file di swap viene scelta **Abilitata, archivia in una cartella personalizzata**, essi vengono creati nella cartella specificata. Questo è utile nei filesystem di rete, per evitare del traffico di rete non necessario.

Salva file di swap ogni

KatePart sincronizza i file di swap sul disco ogni 15 secondi, ma solo se sono cambiati dall'ultima sincronizzazione. Ad ogni modo puoi scegliere un intervallo di sincronizzazione a tuo piacimento.

7.1.4.3 Modi e tipi di file

Questa pagina permette di modificare la configurazione predefinita per documenti aventi specifici tipi MIME. Quando l'editor carica un documento, ne cerca la corrispondenza, tramite le maschere o i tipi MIME, con un tipo di file conosciuto. Se viene trovata più di una corrispondenza verrà usata quella con la più alta priorità.

Tipo di file:

Il tipo di file con la priorità più alta è quello mostrato nella prima casella combinata. Se vengono trovati più tipi di file, verranno mostrati tutti.

Nuovo

Questo viene usato per creare un nuovo tipo di file. Se fai clic su questo pulsante i campi sottostanti si svuotano, quindi possono essere riempiti con i dati appropriati per il nuovo tipo di file.

Elimina

Per eliminare un tipo di file esistente, selezionalo dalla casella combinata e premi il pulsante Elimina.

Proprietà del tipo di file attuale

Il tipo di file con la priorità più alta è quello mostrato nella prima casella combinata. Se vengono trovati più tipi di file, verranno mostrati tutti.

Nome:

Il nome del tipo di file sarà il testo della corrispondente voce del menu. Questo nome è mostrato nel menu **Strumenti** → **Tipi di file**

Sezione:

Il nome della sezione viene usato per organizzare i tipi di file nei menu. Può essere anche usato nel menu **Strumenti** → **Tipi di file**.

Variabili:

Questa stringa permette di configurare le impostazioni di KatePart per i file selezionati da questo tipo MIME usando le variabili di KatePart. Puoi impostare quasi ogni opzione, come l'evidenziazione, la modalità di rientro, eccetera.

Premi **Modifica** per vedere un elenco di tutte le variabili possibili e le loro descrizioni. Spunta la casella a sinistra per abilitare una variabile in particolare, e impostane quindi il valore sulla destra. Alcune variabili forniscono un menu a cascata da cui selezionarne i valori possibili, mentre altre richiedono che tu inserisca manualmente un valore valido.

Per tutte le informazioni su queste variabili, vedi [la configurazione delle variabili dei documenti](#).

Evidenziazione:

Quando crei un nuovo tipo di file, questa casella combinata ti permette di selezionare un tipo di file per l'evidenziazione.

Modalità di indentazione:

Il menu a discesa permette di specificare la modalità di indentazione per i nuovi documenti.

Estensioni dei file:

La maschera per i caratteri jolly permette di selezionare i file per nome. Una tipica maschera usa un asterisco e l'estensione del file, ad esempio *.txt; *.text. La stringa è un elenco di maschere separato da punti e virgola.

Tipi MIME:

Mostra una procedura guidata che consente di scegliere facilmente i tipi MIME.

Priorità:

Imposta una priorità per questo tipo di file. Se uno stesso file è associato a più di un tipo di file, verrà utilizzato quello con priorità più alta.

7.2 Configurazione con le variabili dei documenti

Le variabili di KatePart sono l'implementazione di KatePart delle variabili dei documenti, simili ai file modeline di Emacs e di vi. Nella componente KatePart le righe hanno la seguente forma: **kate: NOME_VARIABILE VALORE; [NOME_VARIABILE VALORE; ...]**. Le righe possono naturalmente essere in un commento, se il formato del file lo prevede. I nomi delle variabili sono costituiti da una sola parola (cioè senza spazi), e tutto quello che c'è fino al successivo punto e virgola è il valore. Il punto e virgola è necessario.

Ecco un esempio di riga di variabili, che impone le regole di rientro per un file C++, Java™ o JavaScript:

```
// kate: replace-tabs on; indent-width 4; indent-mode cstyle;
```

NOTA

Le righe di variabili vengono cercate solo nelle prime e nelle ultime dieci righe.

Inoltre, le variabili dei documenti possono essere messe in un file chiamato `.kateconfig` in qualsiasi cartella, e le impostazioni configurate saranno applicate come se le modeline fossero inserite su ogni file nella cartella e nelle sue sottocartelle. Le variabili dei documenti in `.kateconfig` usano la stessa sintassi delle modeline, ma con delle [opzioni estese](#).

In KatePart ci sono variabili per gestire quasi tutte le configurazioni; inoltre, anche le estensioni possono usare le variabili, il ch  andrebbe documentato nel manuale dell'estensione.

KatePart supporta la lettura delle configurazioni dai file `.editorconfig` se   installata la libreria [editorconfig](#). KatePart cerca automaticamente un `.editorconfig` quando apri un file, pur dando priorit  ai file `.kateconfig`.

7.2.1 Come KatePart usa le variabili

Quando legge la configurazione, KatePart cerca in questi posti (nell'ordine):

- nella configurazione globale;
- nei valori facoltativi di sessione;
- nella configurazione 'Tipi di file';
- nelle variabili del documento in `.kateconfig`;
- nelle variabili del documento presenti nel documento stesso;
- nelle impostazioni modificate in fase di modifica da menu o dalla riga di comando.

Come puoi vedere, le variabili del documento possono essere sostituite solo da modifiche fatte al momento dell'esecuzione. Ogni volta che un documento viene salvato, le sue variabili verranno rilette e sovrascriveranno i cambiamenti fatti dalle voci di menu o dalla riga di comando.

Ogni variabile non elencata qui sotto pu  essere inserita nel documento e ritrovata da altri oggetti, come estensioni, che possono usarle per i propri scopi. Per esempio, la modalit  di rientro variabile usa le variabili dei documenti per la propria configurazione.

Le variabili qui elencate documentano KatePart versione 5.38. In futuro potrebbero essere aggiunte altre variabili. Ci sono tre possibili tipi di valori per le variabili, con le seguenti espressioni valide:

- **BOOL** - on | off | true | false | 1 | 0
- **INT** - un numero intero
- **STRING** - qualsiasi altra cosa

7.2.2 Variabili disponibili

auto-brackets [BOOL]

Abilita l'inserimento automatico delle parentesi.

auto-center-lines [INT]

Imposta il numero di righe autocentranti.

background-color [STRING]

Imposta il colore di sfondo del documento. Il valore deve essere qualcosa leggibile come un colore valido, come `#ff0000`.

backspace-indent [BOOL]

Abilita o disabilita la rimozione del rientro alla pressione di **Backspace**.

block-selection [BOOL]

Attiva o disattiva la [selezione a blocchi](#).

bom | byte-order-mark | byte-order-marker [BOOL]

Attiva o disattiva il marcatore dell'ordine dei byte quando si salvano dei file in formato Unicode (utf8, utf16, utf32).

Da: Kate 3.4 (KDE 4.4)

bracket-highlight-color [STRING]

Imposta il colore per l'evidenziazione delle parentesi. Deve essere qualcosa leggibile come un colore valido, come **#ff0000**.

current-line-color [STRING]

Imposta il colore della riga attuale. Il valore deve essere qualcosa leggibile come un colore valido, come **#ff0000**.

default-dictionary [STRING]

Imposta il dizionario predefinito da usare per il controllo ortografico.

Da: Kate 3.4 (KDE 4.4)

dynamic-word-wrap [BOOL]

Attiva o disattiva l'[andata a capo dinamica](#).

eol | end-of-line [STRING]

Imposta la modalità di fine riga. Le impostazioni valide sono **unix**, **mac** e **dos**.

folding-markers [BOOL]

Attiva o disattiva la visualizzazione degli [indicatori di raggruppamento](#).

folding-preview [BOOL]

Abilita l'anteprima del ripiegamento nei bordi dell'editor.

font-size [INT]

Imposta la dimensione in punti dei caratteri del documento.

font [STRING]

Imposta il carattere del documento. Il valore deve essere un nome di carattere valido, per esempio **courier**.

hl | syntax [STRING]

Imposta l'evidenziazione della sintassi. Le stringhe valide sono tutti i nomi disponibili nel menu. Per esempio, per il C++ basta scrivere **C++**.

icon-bar-color [STRING]

Imposta il colore della barra delle icone. Il valore deve essere qualcosa leggibile come un colore valido, come **#ff0000**.

icon-border [BOOL]

Attiva o disattiva la visualizzazione del bordo delle icone.

indent-mode [STRING]

Imposta la modalità di rientro automatico. Sono riconosciute le opzioni **none**, **normal**, **cstyle**, **haskell**, **lilypond**, **lisp**, **python**, **ruby** e **xml**. Vedi Sezione [3.8](#) per i dettagli.

indent-pasted-text [BOOL]

Attiva o disattiva la regolazione del rientro del testo incollato dagli appunti.

Da: Kate 3.11 (KDE 4.11)

indent-width [INT]

Imposta la larghezza del rientro.

keep-extra-spaces [BOOL]

Imposta se tenere gli spazi aggiuntivi quando si calcola la larghezza del rientro.

line-numbers [BOOL]

Attiva o disattiva la visualizzazione del numero di riga.

newline-at-eof [BOOL]

Aggiungi una riga vuota alla fine del file (EOF) al salvataggio del documento.

Da: Kate 3.9 (KDE 4.9)

overwrite-mode [BOOL]

Attiva o disattiva la modalità sovrascrittura.

persistent-selection [BOOL]

Attiva o disattiva la [selezione persistente](#).

replace-tabs-save [BOOL]

Attiva o disattiva la conversione di tabulazioni in spazi in fase di salvataggio.

replace-tabs [BOOL]

Attiva o disattiva la conversione dinamica di tabulazioni in spazi.

remove-trailing-spaces [STRING]

Rimuove gli spazi finali durante il salvataggio del documento. Le opzioni valide sono:

- **none**, **-** o **0**: non rimuovere mai gli spazi finali.
- **modified**, **mod**, **+** o **1**: rimuovi gli spazi finali solo nelle righe modificate. Le righe modificate vengono evidenziate dal sistema di evidenziazione delle righe.
- **all**, ***** o **2**: rimuovi gli spazi finali in tutto il documento.

scrollbar-minimap [BOOL]

Mostra minimappa di scorrimento.

scrollbar-preview [BOOL]

Mostra l'anteprima nella barra di scorrimento.

scheme [STRING]

Imposta lo schema di colori. Per avere effetto, La stringa deve essere il nome di uno schema di colori presente nella configurazione.

selection-color [STRING]

Imposta il colore della selezione. Il valore deve essere qualcosa leggibile come un colore valido, come **#ff0000**.

show-tabs [BOOL]

Attiva o disattiva il carattere di tabulazione visibile.

smart-home [BOOL]

Attiva o disattiva la [navigazione all'inizio intelligente](#).

tab-indent [BOOL]

Attiva o disattiva il rientro col tasto **Tab**.

tab-width [INT]

Imposta la larghezza a schermo del carattere di tabulazione.

undo-steps [INT]

Imposta il numero di passi di annullamento da memorizzare.

Nota: deprecato da Kate 3 in KDE 4. Questa variabile è ignorata. Il numero massimo di annullamenti è illimitato.

word-wrap-column [INT]

Imposta la larghezza dell'[andata a capo dinamica](#).

word-wrap-marker-color [STRING]

Imposta il colore per l'indicatore di andata a capo. Il valore deve essere qualcosa leggibile come un colore valido, come **#ff0000**.

word-wrap [BOOL]

Attiva o disattiva l'andata a capo statica.

7.2.3 Opzioni estese nei file **.kateconfig**

KatePart cerca sempre un file **.kateconfig** per i file locali (non per quelli remoti), inoltre è possibile impostare delle opzioni basate sui caratteri jolly (estensioni dei file) come segue:

```
kate: tab-width 4; indent-width 4; replace-tabs on;  
kate-wildcard (*.xml): indent-width 2;  
kate-wildcard (Makefile): replace-tabs off;
```

In questo esempio tutti i file usano un valore di `tab-width` ed uno di `indent-width` di 4 spazi, e le tabulazioni sono sostituite con gli spazi. Tuttavia la larghezza di rientro è impostata a 2 spazi per tutti i file `*.xml` e i `Makefile` usano le tabulazioni, cioè queste non sono sostituite dagli spazi.

I caratteri jolly sono separati da virgole, ad esempio puoi specificare più estensioni di file così:

```
kate-wildcard (*.json;*.xml): indent-width 2;
```

Inoltre puoi usare il tipo MIME per associare certi file; ad es. per indentare tutti i file sorgente C con 4 spazi puoi scrivere:

```
kate-mimetype (text/x-c++src): indent-width 4;
```

NOTA

Oltre al supporto nei file **.kateconfig** i caratteri jolly e le variabili dei documenti dipendenti dal tipo MIME sono anche supportati nel file stesso come commenti.

Capitolo 8

Riconoscimenti e licenza

Copyright 2001-2022 di KatePart e KWrite della squadra di Kate.

Basato sull'originale KWrite, che era Copyright 2000 by Jochen Wilhelmy digisnap@cs.tu-berlin.de

Contributi:

- Christoph Cullmann cullmann@kde.org
- Michael Bartl michael.bartl1@chello.at
- Phlip phlip_cpp@my-deja.com
- Anders Lund anders@alweb.dk
- Matt Newell newellm@proaxis.com
- Joseph Wenninger kde@jowenn.at
- Jochen Wilhelmy digisnap@cs.tu-berlin.de
- Michael Koch koch@kde.org
- Christian Gebauer gebauer@kde.org
- Simon Hausmann hausmann@kde.org
- Glen Parker glenebob@nwlink.com
- Scott Manson sdmanson@altel.net
- John Firebaugh jfirebaugh@kde.org
- Nivaldo González nibgonz@gmail.com

La documentazione di KatePart si basa sulla documentazione originale di KWrite, modificata in modo da applicarsi a tutti gli utilizzatori di KatePart.

La documentazione originale di KWrite fu scritta da Thad McGinnis ctmcginis@compuserve.com, con molte modifiche da Christian Tibirna tibirna@kde.org. Fu convertita a docbook e riletta da Lauri Watts lauri@kde.org, e aggiornata da Anne-Marie Mahfouf annma@kde.org e Anders Lund anders@alweb.dk.

La documentazione attuale di KatePart è mantenuta da T.C. Hollingsworth thollingsworth@gmail.com. Per richiedere una modifica, per favore compila una segnalazione di bug nel KDE sistema di tracciamento dei bug, oppure fornisci una correzione nel nostro progetto GitLab.

Manuale di KatePart

Traduzione in italiano di Federico Zenith federico.zenith@member.fsf.org, Marco Poletti, Samuele Kaplun e Luigi Toscano.

Questa documentazione è concessa in licenza sotto i termini della [GNU Free Documentation License](#).

Questo programma è concesso in licenza sotto i termini della [GNU General Public License](#).

Capitolo 9

La modalità di inserimento Vi

Erlend Hamberg

Traduzione della documentazione: Federico Zenith

Aggiornamento della traduzione: Paolo Zamponi

9.1 Modalità di inserimento Vi

Lo scopo della modalità Vi non è quello di essere un sostituto completo di Vim e di supportare tutte le sue funzionalità. Il suo scopo è rendere disponibile la modifica del testo ‘alla Vim’, e le abitudini ad essa legate, ai programmi che usano l’editor di testo KatePart come editor interno.

La modalità Vi punta ad integrarsi bene con il programma, e devia dal comportamento di Vim quando ciò ha senso. Per esempio, `:w` aprirà una finestra di salvataggio nella modalità Vi di KatePart.

Per abilitare la modalità di inserimento Vi per tutte le nuove viste, vai a **Impostazioni** → **Configura KatePart+Modifica** → **Modalità di inserimento Vi**. Da questa scheda puoi impostare le opzioni per la modalità di inserimento Vi, e modificarne la mappatura della tastiera. La modalità di inserimento Vi può essere attivata con l’impostazione **Modalità di inserimento Vi** nel menu **Modifica** (la scorciatoia predefinita è **Meta+Ctrl+V**, dove **Meta** è di norma il tasto **Windows**).

NOTA

Molti comandi da tastiera della modalità Vi sono diversi con le maiuscole, al contrario della maggior parte delle scorciatoie da tastiera di KDE. Ciò vuol dire che **y** e **Y** sono due comandi diversi. Per inserire il comando **y** (copia), assicurati che **Bloc Maiusc** sia disattivato e premi **Y**. Per inserire il comando **Y** (copia fino alla fine della riga), **Shift+Y**.

Ciò non vale per i comandi che usano il tasto **Ctrl**, che possono essere inseriti indipendentemente dalla modalità di **Bloc Maiusc** e senza premere **Shift**. Tuttavia, alcuni comandi richiedono l’uso di combinazioni di un tasto **Ctrl** con un altro tasto in versione maiuscola. Per esempio, per inserire **‘Ctrl+W, h’** (passa alla vista divisa a destra), assicurati che **Bloc Maiusc** sia disattivato, premi **Ctrl+W**, rilascia, e quindi premi **H**.

9.1.1 Incompatibilità con Vim

Ci sono solo alcune poche funzionalità della modalità Vi di KatePart che non sono compatibili con Vim (non contando quelle mancanti). Sono elencate sotto, con le rispettive motivazioni.

- KatePart: **U** e **Ctrl+R** sono il comando rifai.
Vim: **Ctrl+R** è il comando rifai normale, **U** è usato per annullare tutte le ultime modifiche su una riga.
Il motivo di far funzionare **U** come azione di rifai nella modalità Vi di KatePart è che la scorciatoia **Ctrl+R** è normalmente già presa dalla funzione di sostituzione di KatePart (ricerca e sostituzione). Normalmente, la modalità Vi non si sovrapporrà alle scorciatoie di KatePart (questo si può modificare in **Impostazioni** → **Configura KatePart+Modifica** → **Modalità di inserimento Vi**), quindi un'azione di rifai va resa disponibile anche con una 'normale' pressione di un tasto. Inoltre, il comportamento del comando **U** in Vim non si adatta bene al sistema di annullamento interno di KatePart, quindi sarebbe comunque difficile da supportare.
- KatePart: **print** mostra la finestra di **Stampa**.
Vim: **print** stampa le righe dell'intervallo dato, come nel suo antenato ed.
I comandi come **:print** sono disponibili non solo nella modalità Vi, ma anche per gli utenti del KatePart 'normale'; quindi il comando **:print** apre la finestra di stampa, seguendo il principio di minima sorpresa invece di imitare il comportamento di Vim.
- KatePart: **Y** copia fino alla fine della riga.
Vim: **Y** copia tutta la riga, come **yy**.
Il comportamento di Vi per il comando **Y** è in pratica un errore. Sia il comando di modifica che di cancellazione, **c/d**, eseguiranno la loro azione sulla riga attuale, mentre **C/D** funzioneranno dalla colonna del cursore fino alla fine della riga. Tuttavia, sia **yy** che **Y** copiano la riga attuale. Nella modalità Vi di KatePart, **Y** copierà fino alla fine della riga. Ciò è descritto come 'più logico' nella [documentazione di Vim](#).
- KatePart: **O** e **o** aprono [*tot*] nuove righe e ti mettono in modalità di inserimento
Vim: **O** e **o** aprono una nuova riga e inseriscono del testo [*tot*] volte all'uscita dalla modalità di inserimento.
Questo è motivato essenzialmente dalla confusione di molte persone da parte di questo comportamento su un canale IRC su Vim (#vim su Libera Chat).

9.1.2 Cambiare modalità

- La *modalità normale* ti permette di inserire i comandi per navigare o modificare un documento, ed è quella predefinita. Puoi tornarci da qualsiasi altra modalità premendo **Esc**.
- La *modalità visuale* permette di evidenziare del testo in un documento. La maggior parte dei comandi sono validi anche in questa modalità. Puoi attivarla premendo **v** per selezionare i caratteri o **V** per selezionare le righe.
- La *modalità di inserimento* ti permette di modificare il documento direttamente. Puoi attivarla premendo **i** o uno dei vari comandi elencati sotto.
- La *modalità di comando* invoca la riga di comando di KatePart, permettendoti di eseguire molti comandi disponibili nelle implementazioni di Vi oltre ad alcuni specifici di KatePart. Per maggiori informazioni, vedi Sezione 5.2. Per usarla, premi **:**, inserisci il comando e premi **Invio**.

9.1.3 Integrazione con le funzionalità di Kate

- La modalità visuale è attivata automaticamente quando si seleziona del testo col mouse. È attivata anche quando si usano funzioni di Kate che selezionano del testo, come **Seleziona tutto** (dal menu o con **Ctrl+A**).
- Gli indicatori di Vi e i [segnalibri di Kate](#) sono integrati. Quando si crea un indicatore in modalità Vi, viene creato un segnalibro di Kate corrispondente e appare nel menu **Segnalibri**. Allo stesso modo, quando si crea un segnalibro di Kate, viene anche creato un indicatore di Vi alla colonna 0.

9.1.4 Comandi supportati nelle modalità normale e visuale

a	Entra nella modalità di inserimento; aggiungi dopo il cursore
A	Entra nella modalità di inserimento; aggiungi dopo la riga
i	Entra nella modalità di inserimento; aggiungi prima del cursore
Ins	Entra nella modalità di inserimento; aggiungi prima del cursore
I	Entra nella modalità di inserimento; aggiungi prima del primo carattere non vuoto sulla riga
gi	Entra nella modalità di inserimento; aggiungi prima del posto dove si è lasciata l'ultima modalità di inserimento
v	Entra nella modalità visuale; seleziona dei caratteri
V	Entra nella modalità visuale; seleziona delle righe
Ctrl+v	Entra nella modalità visuale; seleziona dei blocchi
gb	Entra nella modalità visuale; riseleziona l'ultima selezione
o	Apri una nuova riga sotto l'attuale
O	Apri una nuova riga sopra l'attuale
J	Unisci righe
c	Modifica: fai seguire un movimento per cancellare ed entrare nella modalità di inserimento
C	Modifica fino alla fine della riga: cancella fino alla fine della riga ed entra in modalità di inserimento
cc	Modifica riga: cancella la riga ed entra nella modalità di inserimento
s	Carattere sostitutivo
S	Riga sostitutiva
dd	Cancella riga
d	Fai seguire un movimento per cancellare
D	Cancella fino alla fine della riga
x	Cancella il carattere alla destra del cursore
Canc	Cancella il carattere alla destra del cursore
X	Cancella il carattere alla sinistra del cursore
gu	Fai seguire un movimento per mettere in minuscolo
guu	Metti la riga attuale in minuscolo
gU	Fai seguire un movimento per mettere in maiuscolo
gUU	Metti la riga attuale in maiuscolo
y	Fai seguire un movimento per 'copiare' (copia)
yy	Copia riga
Y	Copia riga

p	Incolla dopo il cursore
P	Incolla prima del cursore
]p	Incolla dopo il cursore rientrato
[p	Incolla prima del cursore rientrato
r	Fai seguire un carattere per sostituire il carattere dopo il cursore
R	Entra nella modalità di sostituzione
:	Entra nella modalità di comando
/	Cerca
u	Annulla
Ctrl+R	Rifai
U	Rifai
m	Imposta indicatore (può essere usato in seguito dai movimenti)
n	Trova successivo
N	Trova precedente
>>	Fai rientrare la riga
<<	Riduci il rientro della riga
>	Fai rientrare le righe
<	Riduci il rientro delle righe
Ctrl+F	Pagina giù
Ctrl+B	Pagina su
ga	Stampa il valore ASCII del carattere
.	Ripeti l'ultima modifica
==	Allinea la riga
=	Allinea le righe
~	Cambia tra versione maiuscola o minuscola del carattere attuale
Ctrl+S	Dividi la vista orizzontalmente
Ctrl+V	Dividi la vista verticalmente
Ctrl+W, w	Passa alla prossima finestra divisa
Ctrl+W, h CtrlW ←	Passa alla finestra divisa a sinistra
Ctrl+W, l CtrlW →	Passa alla finestra divisa a destra
Ctrl+W, k CtrlW ↑	Passa alla finestra divisa sopra
Ctrl+W, j CtrlW ↓	Passa alla finestra divisa sotto

9.1.5 Movimenti supportati

Questi possono essere usati per spostarsi in un documento nelle modalità normale o visuale, o in congiunzione con uno dei comandi sopracitati. Possono essere preceduti da un conto, che indica quanti movimenti effettuare.

h	Sinistra
←	Sinistra
Backspace	Sinistra
j	Giù

↓	Giù
k	Su
↑	Su
l	Destra
→	Destra
Spazio	Destra
\$	Fine riga
Fine	Fine riga
0	Primo carattere della riga (colonna 0)
&#8598;	Primo carattere della riga
^	Primo carattere non vuoto della riga
f	Fai seguire un carattere da spostare alla destra del cursore
F	Fai seguire un carattere da spostare alla sinistra del cursore
t	Fai seguire un carattere da spostare alla destra del cursore, mettendo il cursore sul carattere precedente
T	Fai seguire un carattere da spostare alla sinistra del cursore, mettendo il cursore sul carattere precedente
gg	Prima riga
G	Ultima riga
w	Parola successiva
W	Parola successiva separata da spazi
b	Parola precedente
B	Parola precedente separata da spazi
e	Fine della parola
E	Fine della parola separata da spazi
ge	Fine della parola precedente
gE	Fine della parola precedente separata da spazi
 	Fai seguire il numero della colonna a cui spostarsi
%	Fai seguire un elemento a cui spostarsi
`	Indicatore
`	Primo carattere non di spaziatura della riga su cui è l'indicatore
[[Quadra aperta precedente
]]	Quadra aperta successiva
[]	Quadra chiusa precedente
] [Quadra chiusa successiva
Ctrl+I	Vai alla posizione successiva
Ctrl+O	Vai alla posizione precedente
H	Vai alla prima riga della schermata
M	Vai al centro della schermata
L	Vai all'ultima riga della schermata
%percentuale	Vai alla percentuale del documento specificata
gk	Sali di una riga visualmente (quando si usa il ritorno a capo dinamico)
gj	Scendi di una riga visualmente (quando si usa il ritorno a capo dinamico)
Ctrl+←	Sposta una parola a sinistra

Ctrl+→	Sposta una parola a destra
---------------	----------------------------

9.1.6 Oggetti di testo supportati

Si possono usare per selezionare certe porzioni di un documento.

iw	Parola interna: parola con spazi inclusi
aw	Una parola: parola senza spazi
i"	Dalle virgolette doppie (") precedenti alle successive, virgolette incluse
a"	Dalle virgolette doppie (") precedenti alle successive, virgolette escluse
i'	Dalle virgolette singole (') precedenti alle successive, virgolette incluse
a'	Dalle virgolette singole (') precedenti alle successive, virgolette escluse
i (Dalla parentesi aperta precedente [(] alla chiusa successiva [)], parentesi incluse
a (Dalla parentesi aperta precedente [(] alla chiusa successiva [)], parentesi escluse
i [Dalla parentesi quadra aperta precedente ([) alla chiusa successiva (]), parentesi quadre incluse
a [Dalla parentesi quadra aperta precedente ([) alla chiusa successiva (]), parentesi quadre escluse
i {	Dalla parentesi graffa aperta precedente ({) alla chiusa successiva (}), parentesi graffe incluse
a {	Dalla parentesi graffa aperta precedente ({) alla chiusa successiva (}), parentesi graffe escluse
i <	Dalla parentesi angolata aperta precedente (<) alla chiusa successiva (>), parentesi angolate incluse
a <	Dalla parentesi angolata aperta precedente (<) alla chiusa successiva (>), parentesi angolate escluse
i `	Dall'accento grave precedente (`) al successivo, accenti inclusi
a `	Dall'accento grave precedente (`) al successivo, accenti esclusi

9.1.7 Comandi supportati nella modalità di inserimento

Ctrl+D	Riduci rientro
Ctrl+T	Fai rientrare
Ctrl+E	Inserisci da sotto
Ctrl+Y	Elimina parola
Ctrl+W	Elimina parola
Ctrl+U	Cancella riga
Ctrl+J	Nuova riga
Ctrl+H	Elimina carattere all'indietro
Ctrl+&#8598;	Vai al primo carattere del documento
Ctrl+R n	Inserisce i contenuti del registro <i>n</i>
Ctrl+O , <i>comando</i>	Entra nella modalità normale per un solo comando
Ctrl+A	Incrementa il numero attualmente selezionato
Ctrl+X	Decrementa il numero attualmente selezionato

9.1.8 L'oggetto di testo tra virgole

Questo oggetto non è presente in Vim. L'oggetto di testo tra virgole facilita la modifica di elenchi di parametri in linguaggi simili al C e in altri elenchi separati da virgole. È sostanzialmente l'area tra due virgole, o tra una virgola e una parentesi. In questa riga mostrata nell'illustrazione sono evidenziati i tre intervalli coperti da questo oggetto di testo.

```
int f(int arg1, double arg2, char arg3);
```

*Intervalli degli oggetti di testo tra virgole. Se il cursore è per esempio su `arg2`, premere **ci** ('cambia tra virgole') eliminerebbe `double arg2` e posizionerebbe il cursore tra le due virgole in modalità di inserimento. Un modo molto comodo per cambiare i parametri di una funzione.*

9.1.9 Funzionalità mancanti

Come già menzionato, l'obiettivo della modalità Vi di KatePart non è supportare il 100% delle funzionalità di Vim.

Appendice A

Le espressioni regolari

Questa appendice contiene introduzione un'essenziale al mondo delle *espressioni regolari*. Documenta le espressioni regolari nella forma disponibile in KatePart, che non è compatibile con quella di perl né con quella disponibile, per esempio, in **grep**.

A.1 Introduzione

Le *espressioni regolari* ci forniscono un modo di descrivere il contenuto richiesto di una stringa di testo in modo comprensibile dal software, al fine di verificare se una sequenza di testo corrisponde al modello fornito; nel caso di applicazioni avanzate, permettono di memorizzare parti del testo corrispondenti.

Un esempio: diciamo che vuoi cercare nel testo tutti i paragrafi che iniziano per uno dei nomi 'Henrik' o 'Pernille' seguiti da una qualche forma del verbo 'say'.

In una ricerca normale cominceresti a trovare il primo nome, 'Henrik', magari seguito da 'sa', così: **Henrik sa**; esaminando le corrispondenze dovresti scartare tutte quelle in cui non si trovano all'inizio di un paragrafo, oltre a quelle in cui le parole che iniziano per 'sa' non sono né 'says' né 'said'. E tutto ciò dovrà essere ripetuto con l'altro nome...

Con le espressioni regolari questo compito può essere svolto con una sola ricerca, e con una precisione più elevata.

Per farlo, le espressioni regolari definiscono delle regole per esprimere nel dettaglio una generalizzazione di una stringa da confrontare. Il nostro esempio, che potremmo esprimere in questo modo: 'Una riga che inizia con 'Henrik' o 'Pernille' (forse preceduti da fino a quattro caratteri di spazio o tabulazione) seguiti da un carattere di spaziatura seguito da 'sa' e poi, o 'ys' o 'id', può essere espresso con la seguente espressione regolare:

```
^[ \t]{0,4}(Henrik|Pernille) sa(ys|id)
```

L'esempio appena introdotto dimostra i quattro concetti alla base delle moderne espressioni regolari, cioè:

- I modelli
- Le asserzioni
- I quantificatori
- I riferimenti all'indietro

Il carattere di accento circonflesso (^) che appare all’inizio dell’espressione è un’asserzione, che è vera solo se il resto della stringa da confrontare è all’inizio di una riga.

Le stringhe `[\t]` e `(Henrik|Pernille) sa(ys|id)` sono modelli. Il primo è una *classe di caratteri* che corrisponde a un carattere di spazio o di tabulazione (orizzontale); l’altro modello contiene prima un sotto-modello che corrisponde a `Henrik o Pernille`, poi un gruppo di caratteri che corrisponde letteralmente a `sa`, ed infine un sotto-modello che corrisponde a `ys o id`.

La sequenza `{0,4}` è un quantificatore che indica ‘da zero a quattro ripetizioni dell’espressione precedente’.

Poiché il software per le espressioni regolari che gestisce il concetto di *riferimento all’indietro* salva l’intera sequenza di caratteri che corrisponde al modello oltre ai sottomodelli racchiusi fra parentesi, se ci fosse modo di accedere a tali riferimenti potremmo mettere le mani sull’intero testo corrispondente al modello (in caso di ricerca di un’espressione regolare in un editor di testo la corrispondenza è indicata come selezionata) oppure al nome trovato, o all’ultima parte del verbo.

Nell’insieme l’espressione corrisponderà a quello che vogliamo, e solo a quello.

Le sezioni seguenti descriveranno in dettaglio come costruire e usare i modelli, le classi di caratteri, le asserzioni, i quantificatori ed i riferimenti all’indietro, mentre la sezione finale fornirà qualche utile esempio.

A.2 I modelli

I modelli consistono in sequenze di caratteri letterali e in classi di caratteri. Possono contenere sotto-modelli, che sono modelli racchiusi fra parentesi.

A.2.1 I caratteri speciali

Nei modelli, così come nelle classi di caratteri, alcuni caratteri hanno un significato speciale. Per abbinare letteralmente qualcuno di questi caratteri, essi devono essere marcati, o resi *di escape*; questo far sapere al software di gestione delle espressioni regolari di trattare tali caratteri col loro significato letterale.

Ciò si fa inserendo come prefisso del carattere una barra inversa (`\`).

Il software per la gestione delle espressioni regolari ignora silenziosamente l’uso di un carattere di escape prima di un carattere senza particolari significati, quindi se viene anteposto, ad esempio, a `j` (`\j`) non è dannoso. Se hai dei dubbi sul fatto che un dato carattere possa avere significati speciali, puoi anteporvi il carattere di escape per sicurezza.

Per inserire un carattere di barra inversa devi ripetere il carattere stesso, scrivendo `\\`.

A.2.2 Classi di caratteri ed abbreviazioni

Una *classe di caratteri* è un’espressione regolare che corrisponde ad uno di un gruppo di caratteri particolare. Nelle espressioni regolari le classi di caratteri sono definite inserendo i caratteri legali per la classe fra parentesi quadre, `[]`, o usando una delle classi predefinite descritte più avanti.

Classi di caratteri semplici contengono solo uno o più caratteri letterali, come ad esempio `[abc]` (che corrisponde ad una delle lettere ‘a’, ‘b’ o ‘c’) o `[0123456789]` (che corrisponde ad una delle cifre decimali).

Poiché le lettere e i numeri hanno un ordine logico, puoi abbreviare le definizioni delle classi che li contengono specificando degli intervalli: `[a-c]` è equivalente a `[abc]`, e `[0-9]` equivale a scrivere `[0123456789]`. Sono ammesse anche le combinazioni di questi costrutti, ad esempio, `[a-fyot1-38]` corrisponde ad un carattere fra ‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’, ‘y’, ‘n’, ‘o’, ‘t’, ‘1’, ‘2’, ‘3’ e ‘8’.

Poiché le lettere maiuscole vengono distinte da quelle minuscole, per creare una classe di caratteri che corrisponda ad 'a' o 'b', senza distinguere fra maiuscole e minuscole, devi scrivere **[aAbB]**.

È anche possibile creare una classe di caratteri 'negativa', che corrisponde ad un 'qualunque carattere tranne quelli elencati'. Per far ciò inserisci un carattere di accento circonflesso '^' all'inizio della classe:

[^abc] corrisponderà ad un carattere qualsiasi *che non sia* 'a', 'b' o 'c'.

Oltre ai caratteri letterali sono definite alcune abbreviazioni, che rendono la vita un po' più semplice:

\a

Questo corrisponde al carattere campanella ASCII (BEL, 0x07).

\f

Corrisponde al carattere di avanzamento carta ASCII (FF, 0x0C).

\n

Corrisponde al carattere ASCII di avanzamento riga (LF, 0x0A, newline in Unix).

\r

Corrisponde al carattere ASCII ritorno carrello (CR, 0x0D).

\t

Corrisponde ad un carattere ASCII di tabulazione orizzontale (HT, 0x09).

\v

Corrisponde ad un carattere ASCII di tabulazione verticale (VT, 0x0B).

\xhhh

Questo corrisponde ad un carattere Unicode che ha la rappresentazione esadecimale hhhh (tra 0x0000 e 0xFFFF). \Ooo (cioè \zero ooo) corrisponde al carattere ASCII/Latin-1 rappresentato dal numero ottale 000 (compreso tra 0 e 0377).

. (punto)

Corrisponde ad un carattere qualsiasi (anche ad un 'nuova riga').

\d

Questo corrisponde ad una cifra decimale. Equivale a [0-9]

\D

Corrisponde ad un carattere non numerico. Equivale a [^0-9] o [^\d]

\s

Corrisponde ad un carattere di spaziatura. Equivale a [\t\n\r]

\S

Corrisponde ad un carattere che non è di spaziatura. Equivale a [^ \t\n\r], ed è uguale a [^\s]

\w

Corrisponde ad un 'carattere di parola' - in questo caso a qualsiasi lettera o numero. Equivale a [a-zA-Z0-9_]

\W

Corrisponde ad un carattere di non-parola - qualsiasi carattere che non sia una lettera, un numero o un carattere di sottolineatura. Equivale a [^a-zA-Z0-9_] o a [^\w]

È anche supportata la *notazione POSIX delle classi*, [**:<nome della classe>:**]; per esempio, [**:numero:**] è equivalente a **\d**, e [**:spazio:**] a **\s**. Vedi [qui](#) la lista completa delle classi di caratteri.

Le classi abbreviate possono essere inserite all'interno di classi personalizzate, ad esempio per richiedere la corrispondenza di n carattere di parola, di uno spazio o di un punto, puoi scrivere **[\w \.]**

A.2.2.1 Caratteri con un significato speciale all'interno delle classi di caratteri

I seguenti caratteri hanno un significato particolare all'interno del costrutto '[' delle classi di caratteri, e devono essere preceduti dal carattere di escape per essere inclusi letteralmente in una classe:

]

Termina la classe di caratteri. Deve essere preceduto dal carattere di escape, a meno che non sia esattamente il primo della classe (può seguire un carattere '^', che indica la negazione della classe).

^ (simbolo di accento circonflesso)

Denota una classe negativa, se è il primo carattere. Deve essere preceduto dal carattere di escape per corrispondere letteralmente se è il primo carattere della classe.

- (trattino)

Denota un intervallo logico di caratteri. Deve sempre essere preceduto dal carattere di escape per essere interpretato alla lettera in una classe di caratteri.

\ (barra inversa)

Il carattere di escape. Deve sempre essere raddoppiato per essere interpretato letteralmente.

A.2.3 Alternative: corrispondenza del tipo 'uno fra'

Se vuoi controllare se il testo corrisponde ad uno fra una serie di modelli alternativi, puoi separare ciascuna alternativa con un carattere | (barra verticale).

Ad esempio, per trovare uno fra 'John' ed 'Harry' devi usare l'espressione **John|Harry**.

A.2.4 I sotto-modelli

I *sotto-modelli* sono modelli racchiusi fra parentesi, ed hanno molti usi nel mondo delle espressioni regolari.

A.2.4.1 Specificare delle alternative

Puoi usare un sotto-modello per raggruppare un insieme di alternative all'interno di un modello più grande. Le alternative sono separate dal carattere '|' (barra verticale).

Ad esempio, per cercare una parola fra 'int', 'float' e 'double', puoi usare il modello **int|float|double**. Se vuoi trovare la parole solo se è seguita da un po' di spazi e delle lettere, inserisci le alternative in un sotto-modello: **(int|float|double)\s+\w+**.

A.2.4.2 Cattura del testo corrispondente (riferimenti all'indietro)

Se vuoi usare un riferimento all'indietro, usa un sotto-modello (**PATTERN**) per ricordare la parte desiderata del modello. Per fare in modo che il sotto-modello non venga invece ricordato, usa un gruppo non di acquisizione (**?:PATTERN**).

Per esempio, se vuoi trovare due ripetizioni della stessa parola separate da una virgola ed eventualmente da qualche spazio bianco puoi scrivere **(\w+),\s*\1**. Il sotto-modello **\w+** troverà una sequenza di caratteri di parola, e l'intera espressione corrisponderà se gli stessi caratteri saranno seguiti da una virgola, zero o più caratteri di spaziatura, e da un'identica sequenza di caratteri di parola (La stringa **\1** si riferisce al *primo sotto-modello racchiuso fra parentesi*).

NOTA

Per evitare ambiguità nell'uso di `\1` quando è seguito da alcune cifre (ad es. `\12` può essere il dodicesimo sotto-modello, o anche il primo con 2), usiamo `\{12}` come sintassi per un sotto-modello con più cifre.

Esempi:

- `\{12}1` vuol dire 'usa il sotto-modello 12'
- `\123` vuol dire 'usa l'1 catturato, quindi 23 come testo normale'

A.2.4.3 Asserzioni di lookahead

Un'asserzione di lookahead è un sotto-modello che inizia con `?=` o con `?!`.

Per esempio, per trovare la stringa letterale 'Bill' ma solo se non è seguita da ' Gates', puoi usare questa espressione: **`Bill(?! Gates)`**. (Verrà trovato 'Bill Clinton' ed anche 'Billy the kid', ma verranno ignorate silenziosamente le altre corrispondenze.)

I sotto-modelli usati per le asserzioni non sono catturati.

Vedi anche [Le asserzioni](#).

A.2.4.4 Asserzioni di Lookbehind

Un'asserzione di lookbehind è un sotto-modello che inizia con `?<=` o con `?<!`.

Lookbehind ha lo stesso effetto di lookahead, ma funziona al contrario. Ad esempio, per abbinare la stringa letterale 'fruit' ma solo quando non preceduta da 'grape' puoi usare questa espressione: **`(?<!grape) fruit`**.

I sotto-modelli usati per le asserzioni non sono catturati.

Vedi anche [Le asserzioni](#).

A.2.5 Caratteri con un significato speciali in un modello

I seguenti caratteri hanno un significato all'interno di un modello, e devi farli precedere da un carattere di escape se vuoi controllarne la presenza letterale:

`\` (barra inversa)

Il carattere di escape.

`^` (simbolo di accento circonflesso)

Asserisce l'inizio della stringa.

`$`

Asserisce la fine della stringa.

`()` (parentesi aperta e chiusa)

Denotano un sotto-modello.

`{ }` (parentesi graffe aperta e chiusa)

Denotano quantificatori numerici.

`[]` (parentesi quadre aperta e chiusa)

Denotano classi di caratteri.

| (barra verticale)

OR logico. Separa le alternative.

+ (segno più)

Quantificatore, uno o più.

*** (asterisco)**

Quantificatore, zero o più.

? (punto interrogativo)

Carattere facoltativo. Può essere interpretato come un quantificatore, zero o uno.

A.3 Quantificatori

I *quantificatori* permettono ad un'espressione regolare di corrispondere ad un numero specificato di caratteri o di caratteri di una classe, o di sotto-modelli.

I quantificatori sono racchiusi da parentesi graffe ({ e }), ed hanno la forma generale {[ripetizioni-minime][, [ripetizioni-massime]]}

L'uso è spiegato meglio con degli esempi:

{1}

Esattamente una ripetizione

{0, 1}

Zero o una ripetizione

{, 1}

La stessa cosa, ma con meno lavoro...

{5, 10}

Almeno cinque ripetizioni, ma meno di dieci.

{5, }

Almeno cinque ripetizioni, nessun massimo.

Inoltre esistono alcune abbreviazioni:

*** (asterisco)**

simile a {0, }, trova un numero qualsiasi di ripetizioni.

+ (segno più)

simile a {1, }, almeno una presenza.

? (punto interrogativo)

simile a {0, 1}, presente al massimo una volta.

A.3.1 Ingordigia

Quando si usano quantificatori senza massimo, le espressioni regolari cercano normalmente di adattarsi all'estensione più lunga di caratteri della stringa esaminata; questo comportamento è normalmente noto come *ingordigia*.

IL software per le espressioni regolari moderno fornisce dei metodi per 'eliminare l'ingordigia', anche se in un ambiente grafico è l'interfaccia a fornire il metodo di accesso a questa funzione. Per esempio, una finestra di ricerca che permette l'uso di espressioni regolari potrebbe avere una casella marcata 'Corrispondenza minimale' per indicare se l'ingordigia è il comportamento predefinito.

A.3.2 Esempi in contesto

Ecco alcuni esempi che usano quantificatori:

`^\d{4,5}\s`

Corrisponde ai numeri in '1234 vai' e '12345 ora', ma non a '567 undici', né a '223459 qualcosa'.

`\s+`

Corrisponde ad uno o più caratteri di spaziatura.

`(b1a){1,}`

Corrisponde a tutta la stringa 'blablabla' e a 'bla' in 'blasone' o in 'cablato'.

`/?>`

Corrisponde a '/'>' in '<closeditem/>' ed anche a '>' in '<openitem>'.

A.4 Le asserzioni

Le *asserzioni* permettono ad un'espressione regolare di corrispondere solo a certe condizioni.

Un'asserzione non corrisponde necessariamente a un carattere, piuttosto impone delle restrizioni sulle condizioni della possibile corrispondenza prima di riconoscerla. Ad esempio, l'asserzione *confine parola* non cerca di trovare un carattere non-parola di fianco ad uno di parola, si assicura solo che non ci sia un carattere parola. Quindi l'asserzione corrisponde dove non c'è un carattere, cioè alla fine della stringa in esame.

Alcune asserzioni hanno effettivamente modelli da confrontare, ma la parte di stringa a cui corrispondono non farà parte del risultato del confronto dell'espressione regolare intera.

Le espressioni regolari qui documentate permettono di usare le seguenti asserzioni:

`^` (accento circonflesso: inizio della stringa)

Corrisponde all'inizio della stringa cercata.

L'espressione **`^Peter`** corrisponderà a 'Peter' nella stringa 'Peter, hey!', ma non in 'Hey, Peter!'

`$` (fine della stringa)

Corrisponde alla fine della stringa cercata.

L'espressione **`you\?$`** corrisponde all'ultimo you della frase 'You didn't do that, did you?' ma non a 'You didn't do that, right?'

`\b` (confine parola)

Corrisponde se c'è un carattere di parola su un lato, ed un carattere che non fa parte di una parola sull'altro.

È utile per trovare le estremità di una parola. Come esempio, all'espressione **`\bin\b`** corrisponde l''in' in 'He came in through the window', ma non l''in' di 'window'.

`\B` (non confine parola)

Corrisponde a tutto quello che non corrisponde a '`\b`'.

Ciò significa che corrisponderà ad esempio all'interno di parole: L'espressione **`\Bin\B`** corrisponde in 'window' ma non in 'integer' o in 'I'm in love'.

(?=MODELLO) (Lookahead positivo)

Un'asserzione di lookahead controlla la parte della stringa che segue una potenziale corrispondenza. Il lookahead positivo impedirà che la stringa corrisponda se il testo che segue non corrisponde al *MODELLO* dell'asserzione, ma il testo a cui corrisponde non sarà incluso nel risultato.

L'espressione **handy** (**?=\\w**) corrisponderà a 'handy' in 'handyman' ma non in 'That came in handy!'

(?!MODELLO) (Lookahead negativo)

Il lookahead negativo impedisce una potenziale corrispondenza se la parte seguente della stringa in esame non corrisponde al *MODELLO*.

L'espressione **const \\w+\\b(?!\\s*&)** corrisponde a 'const char' nella stringa 'const char* foo', mentre non corrisponde a 'const QString' in 'const QString& bar' perché '&' corrisponde al modello dell'asserzione di lookahead negativo.

(?<=MODELLO) (lookbehind positivo)

Un'asserzione lookbehind ha lo stesso effetto di una di lookahead, ma funziona al contrario. Il lookbehind cerca una possibile corrispondenza in una parte della stringa precedente, mentre il lookbehind positivo assocerà una stringa solo se è preceduta dal *MODELLO* dell'asserzione, ma il testo a cui corrisponde non sarà incluso nel risultato.

L'espressione **(?<=cup) cake** corrisponderà a 'cake' se è seguita da 'cup' (in 'cupcake' ma non in 'cheesecake' o solo in 'cake').

(?<!MODELLO) (lookbehind negativo)

Il lookbehind negativo impedisce una potenziale corrispondenza se la parte precedente della stringa in esame non corrisponde al *MODELLO*.

L'espressione **(?<![\\w\\.]) [0-9]+** corrisponderà a '123' nelle stringhe '=123' e '-123', ma non a '123' in '.123' oppure in 'parola123'.

(MODELLO) (Gruppo di cattura)

Il sotto-modello tra parentesi viene catturato e ricordato, in modo che possa venire usato nei riferimenti all'indietro. Per esempio, l'espressione **("+)[^"]*\\1** corrisponde a `""testo""` e a `"testo"`.

Vedi la sezione [Catturare il testo corrispondente \(riferimenti all'indietro\)](#) per ulteriori informazioni.

(?:MODELLO) (Gruppo non di cattura)

Il sotto-modello tra parentesi non viene catturato e non viene ricordato. È preferibile usare sempre un gruppo non di acquisizione se le catture non vengono usate.

Appendice B

Indice analitico

C

commentare, [32](#)

D

decommentare, [32](#)

S

sostituisci, come in sed
cerca, come in sed, [36](#)