Manuale di KatePart

Thad McGinnis
Anne-Marie Mahfouf
Anders Lund
T.C. Hollingsworth
Christoph Cullmann
Lauri Watts
Traduzione della documentazione: Marco Poletti
Traduzione del documento: Samuele Kaplun
:

Indice

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 Opzioni dalla linea di comando

Anche se la maggior parte delle volte si lancia KWrite dal menu dei programmi di KDE o da un'icona del desktop, può anche essere aperto dalla riga di comando in una finestra del terminale. Ci sono diverse opzioni utili disponibili in questa modalità.

Molte altre applicazioni che usano KatePart hanno opzioni da riga di comando simili.

2.2.1 Specifica un file

Specificando il percorso e il nome di un particolare file l'utente può far sì che KWrite apra (o crei) quel file immediatamente all'avvio. Questa opzione può assomigliare a questa:

```
% kwrite
/home/miahome/docs/miofile.txt
```

2.2.2 Specifica un file su Internet

Il metodo sopra menzionato può essere usato anche per aprire file su Internet (se l'utente ha una connessione attiva in quel momento). Ad esempio:

```
% kwrite
ftp://ftp.kde.org/pub/kde/README
```

2.2.3 Altre opzioni dalla linea di comando

Sono disponibili le seguenti opzioni di aiuto dalla riga di comando

`kwrite --help`
Elenca le opzioni di base disponibili dalla riga di comando.

`kwrite --author`
Elenca gli autori di KWrite nella finestra del terminale

`kwrite -v, --version`
Elenca le informazioni sulla versione dell'applicazione.

`KWrite --license`
Visualizza le informazioni di licenza.

`kwrite --desktopfile filename`
Il nome file della voce del desktop per questa applicazione.

`kwrite -e, --encoding encoding URL`
Fa sì che KWrite utilizzi la codifica specificata per il documento.

`kwrite -l, --line line URL`
Si posiziona sulla riga specificata dopo aver aperto il documento.

`kwrite -c, --column column URL`
Si posiziona sulla colonna specificata dopo aver aperto il documento.

`kwrite -i, --stdin`
Fa sì che KWrite legga il contenuto del documento dallo STDIN. Ciò è simile all'opzione - utilizzata comunemente in molti programmi a riga di comando e permette di redirigere l'output di un comando in KWrite.

2.3 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.
Freccia Sinistra	Sposta il cursore a sinistra di un carattere
Freccia Destra	Sposta il cursore a destra di un carattere
Freccia Su	Sposta il cursore in su di una riga
Freccia Giù	Sposta il cursore in giù di una riga
Pag Su	Sposta il cursore in su di una pagina
Alt+Pag Su	Segnalibro precedente
Pag Giù	Sposta il cursore in giù di una pagina
Alt+Pag Giù	Segnalibro successivo

Backspace	Cancella il carattere a sinistra del cursore
Home	Sposta il cursore all'inizio della riga
Fine	Sposta il cursore alla fine della riga
Canc	Cancella il carattere alla destra del cursore (o il testo selezionato)
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.
Shift+Freccia Sinistra	Seleziona un carattere a sinistra nel testo
Shift+Freccia Destra	Seleziona un carattere a destra nel testo
F1	Aiuto
Shift+F1	Che cos'è?
F3	Trova successivo
Shift+F3	Trova precedente
Ctrl+H	Trova selezionato
Ctrl+Shift+H	Trova selezionato all'indietro
Ctrl+A	Seleziona tutto
Ctrl+Shift+A	Deseleziona
Ctrl+Shift+B	Modalità di selezione a blocchi
Ctrl+B	Metti segnalibro
Ctrl+C	Copia il testo selezionato negli appunti.
Ctrl+D	Commenta
Ctrl+Shift+D	Decommenta
Ctrl+F	Trova
Ctrl+G	Va alla riga...
Ctrl+I	Fai rientrare la selezione
Ctrl+Shift+I	Annulla rientro alla selezione
Ctrl+J	Unisci righe
Ctrl+N	Nuovo documento
Ctrl+O	Apre un documento
Ctrl+P	Stampa
Ctrl+Q	Esci - chiude la copia attiva dell'editor
Ctrl+R	Sostituisci
Ctrl+S	Invoca il comando **Salva**.
Ctrl+U	Maiuscolo
Ctrl+Shift+U	Minuscolo
Ctrl+Alt+U	Iniziali maiuscole
Ctrl+V	Incolla il testo degli appunti nella riga di edit.
Ctrl+W	Chiudi
Ctrl+X	Cancella il testo selezionato e lo copia negli appunti.
Ctrl+Z	Annulla
Ctrl+Shift+Z	Rifai
Ctrl+-	Contrai un livello locale
Ctrl+Shift+-	Contrai livello superiore
Ctrl++	Espandi un livello locale

Ctrl+Shift++	Espandi livello superiore
Meta+Ctrl+V	Modalità di inserimento VI
Ctrl+Spazio	Invoca completamento del codice
F5	Ricarica
F6	Mostra bordo per le icone
F7	Passa alla riga di comando
F9	Mostra segni di raggruppamento
F10	A capo automatico dinamico
F11	Mostra i numeri di riga

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 raggruppabili 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:

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↓**, **Home** 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 **Sinistra** e **Destra** 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 **Home** 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, **Pag↑**, **Pag↓**, **Home** e **Fine**, 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 selezionalo, 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 pulsante centrale 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 incrementare 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 casella di testo per inserire lo schema di ricerca.

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** → **Allinea** 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#, Java, 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.

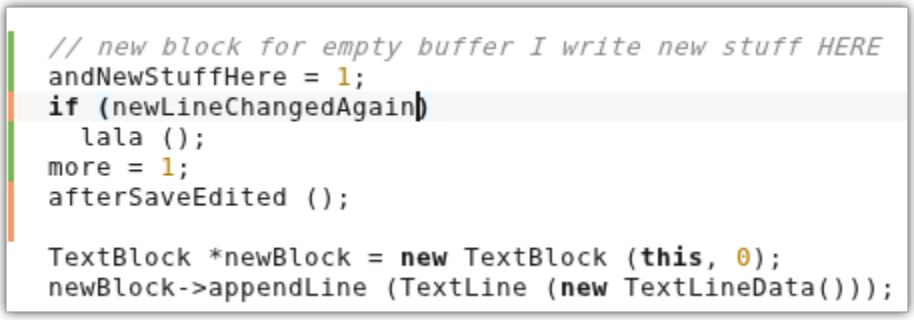
Lisp
: Un sistema di rientro specifico per il linguaggio di scripting Lisp e i suoi dialetti.

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.

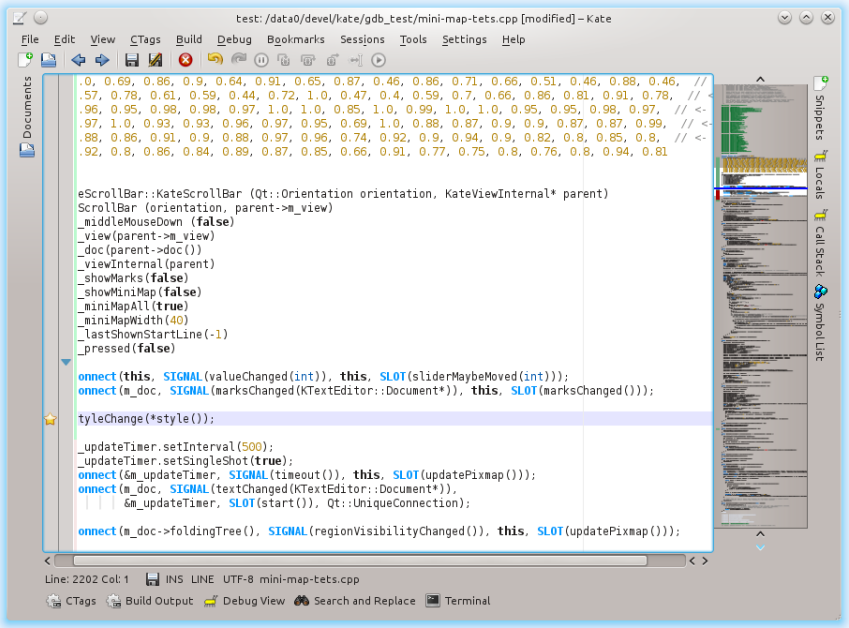


Gli indicatori di modifica delle righe in azione.

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.

Capitolo 4

Voci del menu

4.1 Il menu File

File → Nuovo (Ctrl+N)

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

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 il precedente comando di modifica (digitazione, 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 → 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.

File → Cronologia degli appunti

Questo sottomenu mostrerà l'inizio di porzioni di testo recentemente copiate negli appunti. Seleziona un elemento da questo menu per incollarlo nel file attualmente aperto.

Modifica → Copia come HTML

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

Modifica → Seleziona tutto (Ctrl+A)

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

Modifica → **Deseleziona (Ctrl+Shift+A)**

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

Modifica → **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.

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**.

Modifica → **Trova... (Ctrl+F)**

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 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.

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.

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.

Modifica → **Vai alla parentesi corrispondente (Ctrl+6)**

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

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

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

Modifica → **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.

Modifica → **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.

Modifica → **Va alla riga... (Ctrl+G)**

Apre 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 decrementerà. Si può chiudere la barra facendo clic sul pulsante con un'icona nella parte sinistra.

4.3 Il menu Visualizza

Visualizza → **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.

Visualizza → **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.

Visualizza → **Ingrandisci i caratteri (Ctrl++)**

Aumenta la dimensione del carattere.

Visualizza → **Rimpicciolisci i caratteri (Ctrl+-)**

Diminuisce la dimensione del carattere.

Visualizza → **Schema**

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.

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

Le righe di testo andranno a capo sul bordo della vista nello schermo.

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:

Mostra segni di raggruppamento (F9)

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

Contrai nodo attuale

Contrae la regione contenente il cursore.

Espandi nodo attuale

Espande la regione contenente il cursore.

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.

4.4 Il menu Segnalibri

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.

Segnalibri → **Metti 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).

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.

Segnalibri → **Precedente (Alt+PgUp)**

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.

Segnalibri → **Successivo (Alt+Pag Giù)**

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.

4.5 Il menu Strumenti

Strumenti → **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.

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

Marcando questa opzione si può aggiungere un marcatore esplicito dell'ordine dei byte, per i documenti con codifica unicode. Il BOM (Byte Order Mark) è un carattere Unicode usato per indicare l'ordine dei byte di un file di testo o un flusso di dati. Per maggiori informazioni, vedere Byte Order Mark.

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 → 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** → **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** → **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 → **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 rimpiazzi 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
Apre 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.

Strumenti → **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.

Strumenti → **Allinea**

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

Strumenti → **Commenta (Ctrl+D)**

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

Strumenti → **Decommenta (Ctrl+Shift+D)**

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

Strumenti → **Maiuscolo (Ctrl+U)**

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

Strumenti → **Minuscolo (Ctrl+Shift+U)**

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

Strumenti → **Iniziali maiuscole (Ctrl+Alt+U)**

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

Strumenti → **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.

Strumenti → **Applica il ritorno 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 **Colonna a cui andare a capo** nella scheda Modifica in **Impostazioni** → **Configura editor...**

4.6 I menu Impostazioni e Aiuto

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 rimuovere 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, commentare una selezione che non comprenda completamente l'ultima riga non è possibile.

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

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

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

Per rimuovere gli indicatori di commento, usa il comando **Strumenti** → **Decommenta** oppure la relativa scorciatoia da tastiera; la 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 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, e comandi aggiuntivi sono forniti con le estensioni.

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**, 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; usa il comando **help** per iniziare. 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 **Freccia su** e **Freccia giù**. 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 inoltre abilita 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-ident 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 KatePart, 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
Elimina un passo di rientro delle righe selezionate o della riga 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 (cioè ˝lun˝ ... ˝dom˝).
dddd	Il nome del giorno completo in forma locale (cioè ˝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 (cioè «Gennaio» ... «Dicembre»).
MMM	Il nome del mese abbreviato in forma locale (cioè «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 una 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 *riferimenti 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 pattern 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 nomini 'Miss Jensen' parecchie volte e 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 *riferimenti 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 *riferimenti all'indietro*.
Ottieni: `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 in basso le righe selezionate.

moveLinesUp
Sposta in alto le righe selezionate.

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.

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.

```
##############
# Function: _update_ssn_file
# Print the session data to the file.
sub _update_ssn_file {
.       my $self = shift;
.       my $ssn_file = "$self->{session_vars}->{session_dir}/$self->{session_vars}->{ID}.ssn";
      if ($ssn_file =~ /([\/\w-]+\.ssn)/) { $ssn_file = $1; } # untaint
.       open UPDATE, ">$ssn_file" || die "Couldn't open $ssn_file for writing??, $!\n";
.       foreach $key (keys %{$self}) {
.       .       next if $key eq "session_vars";
.       .       next unless $self->{$key};
.       .       print UPDATE "[$key]\n";
.       .       for (@{$self->{$key}}) { print UPDATE "$_\n"; }
.       }
.       close UPDATE;
}
```

Una funzione Perl, resa con l'evidenziazione della sintassi.

```
###############
# Function: _update_ssn_file
# Print the session data to the file.
sub _update_ssn_file {
.       my $self = shift;
.       my $ssn_file = "$self->{session_vars}->{session_dir}/$self->{session_vars}->{ID}.ssn";
      if ($ssn_file =~ /([\/\w-]+\.ssn)/) { $ssn_file = $1; } # untaint
.       open UPDATE, ">$ssn_file" || die "Couldn't open $ssn_file for writing??, $!\n";
.       foreach $key (keys %{$self}) {
.       .       next if $key eq "session_vars";
.       .       next unless $self->{$key};
.       .       print UPDATE "[$key]\n";
.       .       for (@{$self->{$key}}) { print UPDATE "$_\n"; }
.       }
.       close UPDATE;
}
```

La stessa funzione Perl, senza l'evidenziazione.

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

KatePart nasce con un sistema, flessibile, configurabile e in grado di eseguire l'evidenziazione della sintassi. Nella distribuzione standard sono fornite definizioni per una vasta gamma di linguaggi di programmazione, di scripting e di markup, e per altri formati di file di testo; inoltre puoi 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 assente, 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 si sta utilizzando, e in qualche caso l'autore delle regole della sintassi sarà orgoglioso qualora il 98% del testo venga correttamente renderizzato. Tuttavia, in genere, sarà difficile incontrare quel 2% non corretto.

> **SUGGERIMENTO**
> Puoi scaricare le regole di evidenziazione della sintassi aggiornate o aggiuntive dal sito web di KatePart; fai clic sul pulsante **Scarica i file delle evidenziazioni...** che si trova nella scheda Modi e tipi di file della Finestra di configurazione.

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 quale sia la definizione di sintassi da usare. Mentre stai leggendo il testo del file, e mentre ci 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. Cosicché, se cancelli un carattere che è marcato come inizio o fine di un contesto, lo stile del testo circostante viene cambiato di conseguenza.

Le definizioni di sintassi usate dal sistema di evidenziazione di KatePart sono file XML, che contengono

- 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 sono valutate nell'ordine in cui sono definite, e, qualora l'inizio della stringa corrente coincida con una regola, viene usato il contesto relativo. Il punto di inizio del testo viene spostato nel punto finale in cui la regola è stata applicata, e 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 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 precedente contesto 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 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, e 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 un file sorgente in C++, un sorgente Java™ e un documento HTML in KatePart vedrai che, anche se i formati e le parole scelte per il trattamento speciale sono diversi, i colori utilizzati sono gli stessi. Questo succede perché KatePart ha una lista di stili predefiniti che sono 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 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 definizioni di sintassi fanno usa di altri non predefiniti. Potrebbe valer la pena lanciare la finestra di configurazione se usi spesso un formato, per 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 opera con due: potresti quindi migliorarne l'evidenziazione configurandole in modo che siano leggermente diverse. Tutti gli stili predefiniti disponibili saranno 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 è `$HOME /.local/share`

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

LE SEZIONI PRINCIPALI DEI FILE DI DEFINIZIONE DELL'EVIDENZIAZIONE 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`.

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.

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>
    <contexts>
      <context attribute="Normal Text" lineEndContext="#pop" name=" ↩
          Normal Text" >
        <keyword attribute="Keyword" context="#stay" String="somename" ↩
            />
        <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, il raggruppamento del codice, i commenti e i rientri.

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.

```
<general>
    <comments>
      <comment name="singleLine" start="#"/>
    </comments>
    <keywords casesensitive="1"/>
  </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.

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**.

lineEmptyContext definisce il contesto se viene incontrata una riga vuota. Predefinito: #stay.

fallthrough definisce se il sistema di evidenziazione deve passare al contesto specificato in fallthroughContext qualora nessuna regola venga soddisfatta. Predefinito: *false*.

fallthroughContext specifica il prossimo contesto se nessuna regola viene soddisfatta.

Se **dynamic** è *true* il contesto si ricorderà delle stringhe e dei segnaposti salvati dalle regole dinamiche. Questo serve per esempio nei documenti HERE. 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 StileNumDef, 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.

weakDeliminator 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.

additionalDeliminator definisce delimitatori addizionali.

wordWrapDeliminator 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 e Strumenti → Decommenta. Gli attributi disponibili sono:

name è sia *singleLine* che *multiLine*. Se scegli *multiLine* sono richiesti gli attributi *end* e *region*.

start definisce la stringa usata per iniziare un commento. In C++ sarebbe ˝/*˝.

end definisce la stringa per chiudere un commento. In C++ questo sarebbe ˝*/˝.

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.

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 indentationdel gruppo general definisce quali rientratori verranno usati. Raccomandiamo caldamente però di omettere questo elemento, dato che il rientratore sarà impostato sia definendo un tipo di file, sia aggiungendo un modello di riga al file di testo. Se si specifica un rientratore, infatti, si forzerà l'uso un rientro specifico, che potrebbe non andar bene sempre. Gli attributi disponibili sono:

modo è il nome del rientratore. I rientratori disponibili sono: *normal*, *cstyle*, *haskell*, *lilypond*, *lisp*, *python*, *ruby* e *xml*.

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 \backlash' in Perl, CoffeeScript, e shell, come r'\raw' in Python.

dsSpecialString, SQL, regexes, documenti HERE, modi LaTeX 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.

dsCostant, 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`**).

  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 esempio **`#pop#pop!OtherContext`**.

Alcune regole possono avere delle *regole figlie*, che vengono valutate solo se la regola genitore corrisponde. All'intera stringa corrispondente sarà dato l'attributo definito dalla regola genitore. Una regola con delle regole figlie assomiglia a questo:

```
<RuleName (attributi)>
  <ChildRuleName (attributi) />
  ...
</RuleName>
```

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

ATTRIBUTI COMUNI

- *attribute*: un attributo associato ad un determinato *itemData*.
- *context*: specifica il contesto al quale il sistema di evidenziazione passa se la regola corrisponde.
- *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)*.

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) ↩
    (dynamic) />
```

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

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) (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.

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 chiava da una lista specifica.

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

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 *additionalDeliminator* con l'etichetta *keywords*.

Se una parola chiave da confrontare dovesse contenere un carattere di delimitazione, allora questo dovrebbe essere aggiunto alla proprietà *weakDeliminator* con l'etichetta *keywords*. Questo carattere perderebbe le sue proprietà di delimitatore in tutte le regole *keywords*.

Int

Rileva un numero intero.

```
<Int (attributi comuni) (dynamic) />
```

Questa regola non ha attributi specifici. Le regole figlie sono usate tipicamente per individuare combinazioni di **L** e **U** dopo il numero che sta ad indicare il tipo di intero nel codice programma. Al momento tutte le regole sono consentite come regole figlie, però la DTD permette solo la regola figlia **StringDetect**.

L'esempio seguente associa numeri interi seguiti dal carattere «L».

```
<Int attribute="Decimal" context="#stay" >
  <StringDetect attribute="Decimal" context="#stay" String="L"  ↩
      insensitive="true"/>
</Int>
```

Float

Rileva un numero in virgola mobile.

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

Questa regola non ha attributi specifici. **AnyChar** è permesso come regola figlia e tipicamente usato per rilevare combinazioni, vedi come riferimento la regola **Int**.

HlCOct

Rileva un numero con rappresentazione ottale.

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

Questa regola non ha attributi specifici.

HlCHex

Rileva un numero con rappresentazione esadecimale.

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

Questa regola non ha attributi specifici.

HlCStringChar

Rileva un carattere di escape.

```
<HlCStringChar (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 (\): **abefnr tv´´?**. Inoltre corrisponderanno numeri esadecimali preceduti dal carattere di escape, come per esempio **\xff** e numeri ottali preceduti dal carattere di escape, per esempio **\033**.

HlCChar

Rileva il carattere C

```
<HlCChar (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 HlCStringChar 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

- Se cerchi la corrispondenza di solo due caratteri usa **`Detect2Chars`** invece che **`StringDetect`**. Lo stesso vale per **`DetectChar`**.
- 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:

```
<RegExpr 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.

- 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.

```
<Detect2Chars attribute="Comment" context="#pop" char="*" char1="/"
    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 language.xsd mySyntax.xml**. I file `validatehl.sh` e `language.xsd` si trovano 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 SYSTEM "language.dtd"
[
        <!ENTITY myref    "[A-Za-z_:][\w.:_-]*">
]>
```

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

6.3 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.

6.3.1 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 `%USER%\AppData\Local\katepart5\indentation`. *`%USERPROFILE%`* viene generalmente espanso in `C:\\Users\\utente`.

6.3.1.1 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.

6.3.1.2 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, per esempio 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
    // StrumentiAllinea, 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** → **Allinea**.

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 numbers ≥ 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 allinea 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 `%USER%\AppData\Local\katepart5\indentation`. *`%USER%`* viene generalmente espanso in `C:\\Users\\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 contribuirli al progetto KatePart contattando la lista di distribuzione.

6.3.2 Script da riga di comando

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 `%USER%\AppData\Local\katepart5\commands`. *`%USER%`* viene generalmente espanso in `C:\\Users\\user`.

6.3.2.1 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.

6.3.2.2 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, per esempio 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 i18n("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**.

6.3.2.2.1 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 contribuirli al progetto KatePart contattando la lista di distribuzione.

6.3.3 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_DIRS /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, per esempio 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 `%USER%\AppData\Local\katepart5\libraries`. *`%USER%`* viene generalmente espanso in `C:\\Users\\user`.

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`.

6.3.3.1 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.

6.3.3.1.1 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(altro);`

`Cursor Cursor.clone();`

Restituisce un clone del cursore.

Esempio: `var clone = cursor.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)`'.

6.3.3.1.2 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))`'.

6.3.3.2 Funzioni globali

Questa sezione elenca tutte le funzioni globali.

6.3.3.2.1 Leggere e includere file

`String read(String file);`

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

`void require(String file);`

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

Da: KDE 4.10

6.3.3.2.2 Debug

`void debug(String testo);`

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

6.3.3.2.3 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, e così via.

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 e così via.

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 e così via.

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* 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 e così via.

6.3.3.3 L'API delle viste

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.

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.

6.3.3.4 L'API dei documenti

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 del documento, o il tipo di 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` 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 documentEnd();

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`.

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.

Si può invocare il sistema di **Aiuto**, accettare le impostazioni correnti e chiudere la finestra di dialogo con il pulsante **OK** o, 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

A capo automatico dinamico

Se quest'opzione è marcata, le righe proseguiranno a capo quando raggiungeranno il bordo della finestra sullo schermo.

Indicatori di a capo automatico dinamico (se applicabile)

Scegli quando mostrare gli indicatori di a capo automatico dinamico, **Spenti**, **Dopo i numeri di riga** o **Sempre attivi**.

Allinea le righe mandate a capo dinamicamente alla profondità di rientro:

Abilita l'allineamento verticale delle righe mandate a capo dinamicamente al livello di rientro della prima riga. Ciò può aiutare a rendere il codice e i marcatori più leggibili.

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 seguenti righe andate a capo automaticamente.

Evidenziazione degli spazi

Evidenzia le tabulazioni
L'editor mostra un simbolo » per indicare la presenza di una tabulazione nel testo.

Evidenziazione degli spazi finali
L'editor mostrerà dei punti per indicare la presenza di spazi finali in fondo alle righe.

Dimensione marcatore di evidenziazione
Usa il cursore per modificare la dimensione del marcatore di evidenziazione visibile.

Avanzate

Mostra righe di rientro
Se marcata, l'editor mostrerà delle righe verticali per favorire l'identificazione delle righe rientrate.

Evidenzia intervallo tra parentesi selezionate
Se abilitato, l'intervallo tra le parentesi selezionate corrispondenti sarà evidenziato.

Anima le parentesi corrispondenti
Se abilitato, un passaggio sopra le parentesi ({, [,], },(or)) animerà rapidamente la parentesi corrispondente.

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.

7.1.1.2 Bordi

Bordi

Mostra segni di raggruppamento
Se questa opzione è marcata, la vista corrente mostrerà degli indicatori per il raggruppamento del codice, se il raggruppamento del codice è disponibile.

Mostra l'anteprima del codice ripiegato
Se marcata, l'anteprima del testo ripiegato viene mostrata in una finestra a comparsa al passaggio sopra ad una regione ripiegata

Mostra bordo per le icone
Se marcata, 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 marcata, sul lato sinistro saranno visualizzati i numeri di riga.

Mostra segni di modifica delle righe
Se attivata, saranno visibili i segni di modifica delle righe. Per maggiori informazioni, vedi Sezione 3.9.

Mostra i segni nella barra di scorrimento
Se questa opzione è marcata la vista attuale mostrerà indicatori sulla barra di scorrimento verticale. Questi segni possono mostrare, ad esempio, i segnalibri.

Mostra l'anteprima del testo nella barra di scorrimento
Se questa opzione è marcata 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.

Mostra minimappa di scorrimento

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, disattivale 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 creazione

Ogni nuovo segnalibro verrà aggiunto in coda, indipendentemente da dove è collocato nel documento.

Per posizione

I segnalibri saranno ordinati a seconda dei numeri di riga a cui sono collocati.

7.1.2 Caratteri e colori

Questa sezione della finestra permette di configurare tutti i caratteri e i colori in qualsiasi schema di colori, oltre a creare nuovi schemi o eliminare quelli esistenti. Ciascuno schema ha impostazioni per i colori, i tipi di carattere e per gli stili di evidenziazione del testo.

KatePart preselezionerà lo schema attualmente attivo. Se si vuole lavorare su uno schema diverso si inizia selezionandolo dalla casella combinata **Schema**. Con i pulsanti **Nuovo** e **Elimina** si possono creare nuovi schemi e cancellare quelli esistenti.

Nella parte bassa della pagina si può selezionare lo **schema predefinito per KatePart**.

Normalmente, KatePart baserà il suo schema di colori su quello attuale di KDE. Puoi riportare un singolo colore al suo valore predefinito facendo clic sulla freccia di azzeramento a destra della sua voce nell'editor dei colori, o puoi riportare tutti i colori ai loro valori predefiniti facendo clic su **Usa lo schema di colori di KDE** in fondo al pannello.

> SUGGERIMENTO
> Puoi regolare lo schema di colori di KDE nel modulo Colori delle Impostazioni di sistema.

7.1.2.1 Colori

Colori di sfondo dell'editor

Area di testo

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

Testo selezionato

Il colore di sfondo del testo selezionato. Il valore predefinito è il colore globale della selezione, così come è configurato dalle preferenze dei colori di KDE.

Riga attuale

Imposta il colore della riga attuale. Se è un po' diverso da quello del testo normale, permette di tenere l'attenzione sulla riga.

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

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

Bordo delle icone

Area di sfondo
Questo colore è 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.

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

Segno di ritorno a capo
Questo colore è 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.

Raggruppamento del codice
Questo colore è 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 del raggruppamento del codice.

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

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

Linea degli errori di ortografia
Questo colore si usa per indicare gli errori di ortografia.

Segni di tabulazione e spazio
Questo colore è usato per gli indicatori di spazi bianchi, quando sono abilitati.

Linea di rientro
Questo colore viene usato per disegnare una linea alla sinistra dei blocchi rientrati, se questa funzionalità è abilitata.

Evidenziazione delle parentesi
Questo colore è usato per lo sfondo delle parentesi corrispondenti.

Colori degli indicatori

Segnalibro
Questo colore viene usato per indicare i segnalibri. Per maggiori informazioni, vedi Sezione 3.6.

Punto d'interruzione attivo
Questo colore viene usato dall'estensione per GDB per indicare un punto d'interruzione attivo. Per maggiori informazioni, vedi la documentazione dell'estensione per GDB.

Punto d'interruzione raggiunto
Questo colore viene usato dall'estensione per GDB per indicare un punto d'interruzione raggiunto durante il debugging. Per maggiori informazioni, vedi la documentazione dell'estensione per GDB.

Punto d'interruzione disabilitato
Questo colore viene usato dall'estensione per GDB per indicare un punto d'interruzione inattivo. Per maggiori informazioni, vedi la documentazione dell'estensione per GDB.

Esecuzione
Questo colore viene usato dall'estensione per GDB per indicare la riga attualmente in esecuzione. Per maggiori informazioni, vedi la documentazione dell'estensione per GDB.

Avviso
Questo colore viene usato dall'estensione di generazione per indicare una riga che ha causato un avviso del compilatore. Per maggiori informazioni, vedi la documentazione dell'estensione di generazione.

Errore
Questo colore viene usato dall'estensione di generazione per indicare una riga che ha causato un errore del compilatore Per maggiori informazioni, vedi la documentazione dell'estensione di generazione.

Modelli e frammenti di testo

Sfondo
Questo colore è usato dall'estensione dei frammenti di Kate per colorare lo sfondo di un frammento.

Segnaposto modificabile
Questo colore viene usato dall'estensione dei frammenti di Kate per segnare un segnaposto modificabile con un clic.

Segnaposto modificabile attivo
Questo colore viene usato dall'estensione dei frammenti di Kate per segnare il segnaposto che stai attualmente modificando.

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.

Usa lo schema di colori di KDE
Fare clic su questo pulsante imposterà tutti i colori sopra definiti secondo lo schema di colori attualmente definito nelle Impostazioni di sistema di KDE. Per maggiori informazioni, vedi la documentazione del modulo di controllo dei colori di KDE.
Se non usi gli spazi di lavoro Plasma di KDE, questo pulsante non avrà effetto, e potrebbe non essere presente.

7.1.2.2 Carattere

Qui si può scegliere il tipo di carattere dello schema. Si può scegliere fra tutti i tipi di carattere disponibili nel sistema, ed impostare la dimensione predefinita. Un testo di esempio è mostrato nella parte inferiore della finestra, per vedere gli effetti delle modifiche.

Per maggiori informazioni su come selezionare un carattere, vedi la sezione Scegliere i caratteri della documentazione sui Fondamentali di KDE.

7.1.2.3 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: per esempio, il testo dei commenti è uguale in quasi tutti i formati di testo che Kate 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.

7.1.2.4 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 deimpostare 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.

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

7.1.3 Modifica

7.1.3.1 Generale

A capo automatico statico

L'a capo automatico è una funzionalità che fa iniziare automaticamente all'editor una nuova riga di testo e sposta il cursore all'inizio di quella nuova riga. KatePart comincerà automaticamente una nuova riga di testo quando la riga corrente raggiungerà la lunghezza specificata dall'opzione Colonna a cui andare a capo.

Abilita a capo automatico statico

Attiva o disattiva l'a capo automatico statico.

Mostra indicatore di a capo statico (se applicabile)

Se questa opzione è marcata, 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.

Colonna a cui andare a capo:

Se l'opzione Abilita a capo automatico statico è marcata questa casella determina la lunghezza (in caratteri) alla quale l'editor comincerà automaticamente una nuova riga.

Modalità di inserimento

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 automatiche

Quando l'utente digita una parentesi sinistra ([, (, o {) KatePart inserirà automaticamente la parentesi destra (},) o]) alla destra del cursore.

Quando c'è del testo selezionato, la digitazione di uno dei caratteri spezza il testo selezionato.

Copia e incolla

Copia o taglia la riga attuale se non c'è una selezione

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.

7.1.3.2 Navigazione del testo

Movimento del cursore di testo

Home e Fine intelligenti

Se è marcato, 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 Su** o **Pag Giù**. 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.

Centratura 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.

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 testo **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 marcata, 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 marcata, il testo incollato dagli appunti sarà fatto rientrare. Si può rimuovere il rientro con il comando **Annulla**.

Azioni di rientro

Il tasto Backspace nello spazio vuoto iniziale toglie il rientro
Se è marcato, 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 si vuole che **Tab** allinei la riga attuale nel blocco di codice attuale come in emacs, si può rendere **Tab** una scorciatoia per l'azione **Allinea**.

Avanza sempre alla prossima tabulazione
Se è marcato, 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 marcato, il tasto **Tab** farà rientrare sempre la riga corrente del numero di posizioni specificato in **Larghezza di rientro**.

Aumenta il livello di rientro se in uno spazio vuoto iniziale

Se è marcato, il tasto **Tab** farà rientrare la riga corrente 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.

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. Si può scegliere tra UNIX®, DOS/Windows®o Macintosh.

Rilevamento automatico delle fini di riga
Si può per far riconoscere automaticamente all'editor 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.

7.1.4.2 Avanzate

Copia di sicurezza al quando si salva

La copia di sicurezza al salvataggio fa copiare a KatePart il file su disco in <prefisso><nomefile><suffisso> prima di salvare le modifiche. Il suffisso predefinito è ~ ed il prefisso è vuoto.

File locali

Marcare la casella se si vuole una copia di sicurezza dei file locali quando si salva.

File remoti

Marcare la casella se si vuole una copia di sicurezza dei file remoti quando si salva.

Prefisso

Inserisci il prefisso per i nomi delle copie di sicurezza.

Suffisso

Inserisci il suffisso per i nomi delle copie di sicurezza.

Opzioni dei file di swap

KatePart può recuperare (la maggior parte di) ciò che è stato scritto dall'ultimo salvataggio in caso di errore o caduta di corrente. Un file di swap (`.swp.<nome_file>`) viene creato alla prima azione di modifica di un documento. Se l'utente non salva le modifiche e KatePart si impianta, il file di swap rimane sul disco. All'apertura di un file, KatePart controlla se c'è un file di swap associato, e in tal caso chiede all'utente se vuole recuperare i dati persi o meno. L'utente ha anche la possibilità di vedere le differenze tra il file originale e quello recuperato. Il file di swap viene eliminato dopo ogni salvataggio e all'uscita normale 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 abilitato, i file di swap sono salvati nella stessa cartella del file. Se viene scelta una **Cartella alternativa** i file di swap sono creati nella cartella specificata. Questo è utile nei filesystem di rete, per da evitare traffico di rete non necessario.

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 si usa per creare un nuovo tipo di file. Dopo aver fatto clic su questo pulsante, i campi sottostanti si svuotano e si possono riempire con i dati appropriati per il nuovo tipo di file.

Elimina
Per eliminare un tipo di file esistente, lo si seleziona dalla casella combinata e si preme 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 in **Strumenti** → **Tipi di file**

Sezione:
Il nome della sezione è 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 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 per selezionarne i valori possibili, mentre altre permettono di inserire un valore valido manualmente.
Per tutte le informazioni su queste variabili, vedi la configurazione delle variabili dei documenti.

Evidenziazione:
Quando si crea un nuovo tipo di file, questa casella combinata 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.

Scarica i file delle evidenziazioni...
Fai clic su questo pulsante per scaricare descrizioni delle sintassi nuove o aggiornate dal sito Web di KatePart.

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 ché andrebbe documentato nel manuale dell'estensione.

KatePart supporta la lettura delle configurazioni dai file `.editorconfig` se è installata la libreria editorconfig. KatePart cerca automaticamente un `.editorconfig` quando apri un file, pur dando priorità ai file `.kateconfig`.

7.2.1 Come KatePart usa le variabili

Quando legge la configurazione, KatePart cerca i questi posti (nell'ordine):

- la configurazione globale;
- valori facoltativi di sessione;
- la configurazione 'Tipi di file';
- variabili del documento in `.kateconfig`;
- variabili del documento presenti nel documento stesso;
- 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 con voci di menu o dalla riga di comando.

Ogni variabile non elencata qui sotto può essere inserita nel documento e ritrovata da altri oggetti, come estensioni, che possono usarle per i propri scopi. Per esempio, la modalità 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-indents` [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.

Da: KDE 4.10.

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-indents` [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 esempio 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

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

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 ctmcginnis@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 responsabilità di T.C. Hollingsworth tchollingsworth@gmail.com. Per piacere, invia i tuoi commenti e suggerimenti alla lista di distribuzione di sviluppo di KatePart a kwrite-devel@kde.org o apri una richiesta sul sistema di tracciamento degli errori di KDE.

Traduzione in italiano di Federico Zenith federico.zenith@member.fsf.org, Marco Poletti, Samuele Kaplun e Luigi Toscano.

Capitolo 9

La modalità di inserimento Vi

Erlend Hamberg
Traduzione della documentazione: Federico Zenith

9.1 Modalità di inserimento Vi

Lo scopo della modalità Vi non è di essere un sostituto completo di Vim e 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 normalmente 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 con 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 per il comando di modifica che di cancellazione, **cc**/**dd** 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 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 freenode).

9.1.2 Cambiare modalità

- La *modalità normale* permette di inserire 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 caratteri o **V** per selezionare righe.
- La *modalità di inserimento* permette di modificare il documento direttamente. Puoi attivarla premendo **i** o uno dei diversi comandi elencati sotto.
- La *modalità di comando* invoca la riga di comando di KatePart, permettendo 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
Inserisci	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 caratteri
V	Entra nella modalità visuale; seleziona righe
Ctrl+v	Entra nella modalità visuale; seleziona blocchi
gb	Entra nella modalità visuale; riseleziona l'ultima selezione
o	Apre una nuova riga sotto l'attuale
O	Apre una nuova riga sopra l'attuale
J	Unisci righe
c	Modifica: fai seguire un movimento per cancellare e 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 e 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
Elimina	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
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)
Home	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**	Salta in avanti alla posizione successiva
Ctrl+**O**	Salta indietro 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 specificata del documento
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

Questi 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

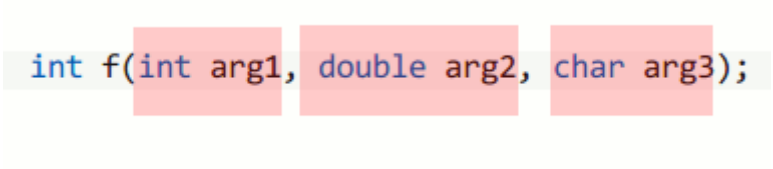
9.1.7 Comandi supportati nella modalità di inserimento

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+Home	Vai al primo carattere del documento
Ctrl+R n	Inserisce i contenuti del registro `n`
Ctrl+O, `comando`	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.



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'obbiettivo 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-fynot1-38]`** 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).

\xhhhh
: Questo corrisponde ad un carattere Unicode che ha la rappresentazione esadecimale hhhh (tra 0x0000 e 0xFFFF). \Oooo (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. Nota che il carattere di sottolineatura (_) non viene considerato un carattere di parola, come succede nelle espressioni regolari di perl. Equivale a `[a-zA-Z0-9]`

\W
: Corrisponde ad un 'carattere di non-parola' - qualsiasi carattere che non sia una lettera o un numero. Equivale a `[^a-zA-Z0-9]` o a `[^\w]`

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 \.]`**

> NOTA
> La notazione POSIX per le classi, **`[:<nomeclasse>]`** non è al momento supportata.

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, per fare in modo che la sezione che ti interessa del modello sia ricordata.

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.2.5 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

`(bla){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?' man 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.

Appendice B

Indice analitico