

YATE - Yet Another Text Editor

Benutzerdokumentation

Einleitung

yate (*yet another text editor*) ist ein einfacher Texteditor, der neben einfachen Funktionen zur Bearbeitung von Texten auch Funktionen zur Verfügung stellt, die den Benutzer bei der Bearbeitung von Programmcode unterstützen. Dazu ermöglicht yate die Verwaltung von mehreren Quelldateien in Projekten, die gemeinsam gespeichert und geladen werden können, und für die projektabhängige Einstellungen gespeichert werden können. Neben der Projektverwaltung bietet der Editor außerdem für einige der am weitesten verbreiteten Programmiersprachen ein Syntaxhighlighting an, das den Programmierer bei der Arbeit unterstützt. Die Farben, die für das Highlighting eingesetzt werden, können individuell konfiguriert und bei Bedarf in einer Projektkonfiguration gespeichert werden.

Übersicht

Die Oberfläche von yate ist in mehrere Bereiche gegliedert. Die folgende Abbildung zeigt ein Beispiel dafür, wie das Programm während dem Betrieb aussieht:

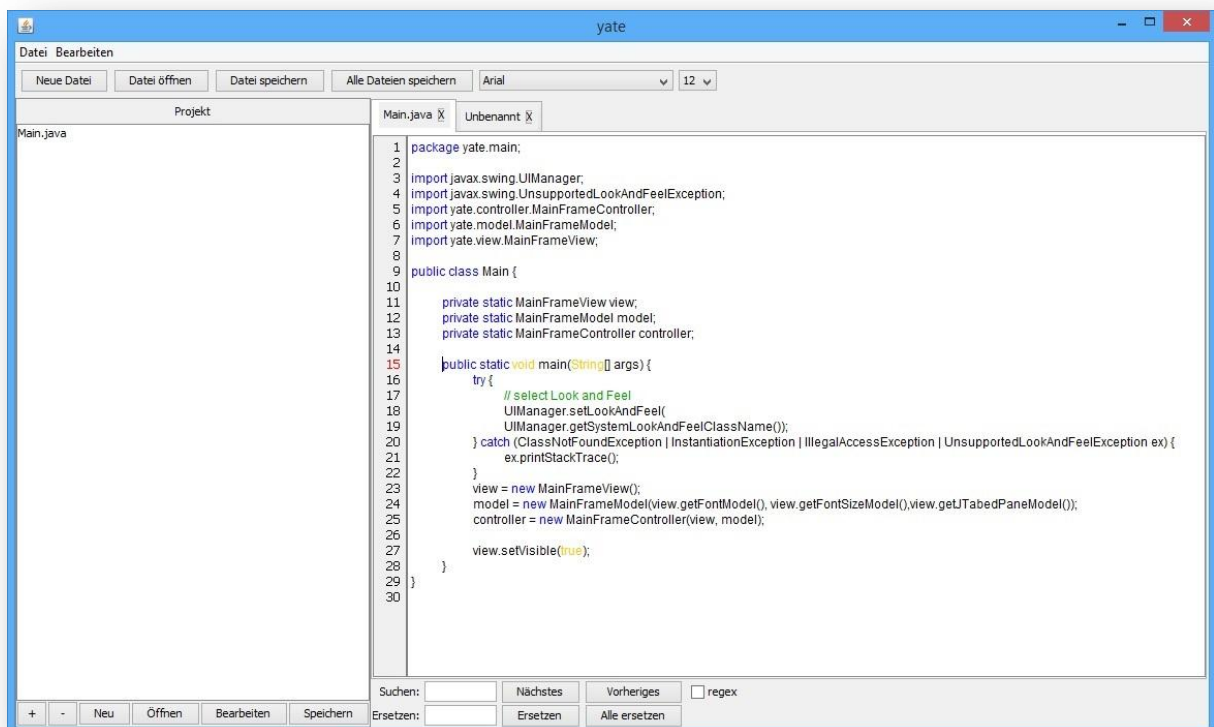


Abbildung 1: Übersicht über die Oberfläche

Den zentralen Bereich bildet der Eingabebereich, in dem der gerade bearbeitete Text dargestellt wird. Am linken Rand der Oberfläche befindet sich das Projektmenü. Hier werden der Name des aktuellen Projekts, die Liste der dazugehörigen Dateien, sowie einige Buttons angezeigt, die

Funktionen zur Verwaltung der Projekte anbieten. Im unteren Bereich der Oberfläche, direkt unter dem Eingabebereich, befinden sich Funktionen, die zur Suche im Text, sowie für das Ersetzen von bestimmten Begriffen im Text dienen.

Über dem Eingabebereich sind die zurzeit geöffneten Dateien in Form von Tabs organisiert. Am oberen Rand der Oberfläche befinden sich Funktionen zum Speichern und Öffnen von Dateien, sowie zum Einstellen der Ansicht, wie z.B. der Schriftgröße oder der Schriftart. Außerdem werden über ein weiteres Menü Optionen zur Auswahl der genutzten Programmiersprache und dem Einstellen der Highlighting-Farbe angeboten.

Bearbeiten von Dateien

Der Kernaspekt eines Texteditors ist die Bearbeitung von Textdateien. Aus diesem Grund bietet *yate* die Möglichkeit, Dateien zu öffnen, zu Speichern und parallel zu Bearbeiten. Außerdem stehen einige Basisfunktionen zur Verfügung, um das Aussehen des Bearbeiteten Texts anzupassen, und um die Orientierung im Text zu erleichtern.

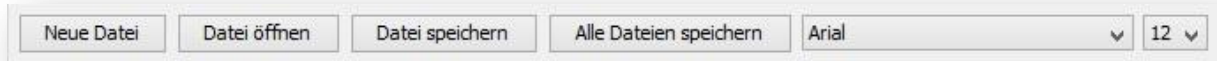


Abbildung 2: Menü zum Öffnen und Speichern von Dateien

Speichern und Öffnen von Dateien

Zum Speichern und Öffnen von Dateien stehen mehrere Funktionen zur Verfügung.

Neue Datei

Über diesen Button kann eine neue Datei erstellt werden. Diese wird zunächst ohne Namen in einem neuen Tab geöffnet. Standardmäßig ist für diese Datei die Programmiersprache Java ausgewählt, sie kann allerdings später über das Menü angepasst werden.

Datei öffnen

Dieser Button dient zum Öffnen einer bereits vorhandenen Datei. Nach der Auswahl der zu öffnenden Datei wird diese in einem neuen Tab angezeigt und der Inhalt der Datei wird geladen. Beim Öffnen versucht *yate*, über die Dateierendung der geöffneten Datei zu ermitteln, welche Programmiersprache voreingestellt wird (siehe Kapitel "*Syntaxhighlighting*"). Kann keine Sprache ermittelt werden, wird Java als Standardsprache voreingestellt.

Datei speichern/Alle Dateien speichern

Mithilfe des Buttons "*Datei speichern*" lässt sich die Datei, die in dem aktuell fokussierten Tab geöffnet ist, speichern. Falls die Datei bei der Betätigung des Buttons noch keinen gültigen Namen hat, kann in einem Dialog ein Name und ein Speicherort festgelegt werden. Andererseits werden Änderungen mit den zuvor festgelegten Dateiinformationen gespeichert.

Über den Button "*Alle Dateien speichern*" werden alle zurzeit geöffneten Dateien gespeichert. Auch hier findet für jede Datei eine Abfrage statt, falls für sie noch keine Dateiinformationen hinterlegt worden sind.

Schriftart festlegen

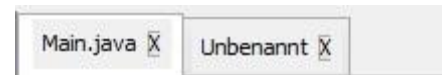
Über das Auswahlmenü können für die Anzeige verschiedene Schriftarten festgelegt werden. Die verfügbaren Schriftarten orientieren sich am Betriebssystem, auf dem das Programm ausgeführt wird. Die Einstellung für die Schriftart wird beim Neustart des Programms verworfen.

Schriftgröße festlegen

Über dieses Auswahlmenü kann, ähnlich wie bei der Schriftart, festgelegt werden, in welcher Schriftgröße der Text dargestellt werden soll. Genau wie bei der Schriftart werden auch diese Einstellungen beim Neustart des Programms verworfen.

Bearbeiten von mehreren Dateien

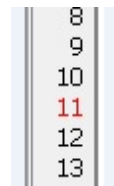
Zur Bearbeitung von mehreren Dateien parallel bietet *yate* eine Leiste über dem Bearbeitungsbereich an, über die zwischen mehreren geöffneten Dateien navigiert werden kann.



Innerhalb der Tabs wird der Dateiname angezeigt. Der Dateiname "*Unbenannt*" ist der Standardname für neue Dateien, für die noch keine Dateiinformationen wie Dateiname bzw. Speicherort hinterlegt wurden. Durch einen Klick auf einen Tab wird diese fokussiert und der Inhalt der Datei wird im Bearbeitungsbereich angezeigt. Mit einem Klick auf das "X"-Symbol wird ein Tab geschlossen.

Zeilenanzeige

Zur besseren Orientierung im Code zeigt eine Leiste links neben dem Text die Zeilennummern an. Die aktuelle Zeile wird dabei farbig hervorgehoben.



Suchen und Ersetzen

Über das Menü unter dem Eingabebereich bietet *yate* die Möglichkeit, das aktuelle Dokument zu durchsuchen. Zusätzlich ist es möglich, gefundene Vorkommen des gesuchten Begriffs zu ersetzen. Über ein Kennzeichen kann außerdem festgelegt werden, dass neben einfachen Begriffen auch reguläre Ausdrücke gesucht werden können.



Abbildung 3: Menü zum Suchen und Ersetzen

Suchen

In diesem Textfeld wird der zu suchende Begriff bzw. reguläre Ausdruck eingegeben.

Ersetzen

In diesem Textfeld wird der Text eingegeben, durch den die gefundenen Vorkommen ersetzt werden sollen.

Nächstes/Vorheriges

Sucht das nächste bzw. vorherige Vorkommen innerhalb des Texts. Alle gefundenen Vorkommen werden mit einer gelben Hintergrundfarbe hinterlegt. Das aktuelle Element wird dagegen mit grün hervorgehoben.

Ersetzen/Alle Ersetzen

Durch den Button "Ersetzen" wird das aktuell gefundene Element (grün hervorgehoben) mit dem vorgegebenen Text ersetzt. Anschließend wird das nächste Element fokussiert. Mithilfe des Buttons "Alle Ersetzen" können automatisch alle gefundenen Vorkommen ersetzt werden.

regex

Ist dieses Kennzeichen gesetzt, werden sowohl bei der Standardsuche, als auch bei der Suche zum Ersetzen von Vorkommen, reguläre Ausdrücke unterstützt.

```
public class Main {
    private static MainFrameView view;
    private static MainFrameModel model;
    private static MainFrameController controller;

    public static void main(String[] args) {
        try {
            // select Look and Feel
        }
    }
}
```

Abbildung 4: Beispiel: Suche nach "private"

Syntaxhighlighting

Allgemein

Das Syntaxhighlighting erleichtert dem Anwender die Bearbeitung von Programmcode innerhalb des Editors. Dazu werden, abhängig von der ausgewählten Programmiersprache, verschiedene Schlüsselwörter, Literale, Kommentare und ähnliche zentrale Strukturen der jeweiligen Programmiersprache farbig hervorgehoben. So ist es dem Anwender möglich, die Funktion von bestimmten Codebereichen bereits anhand ihrer Farbe zu identifizieren, ohne den Code selbst überhaupt zu lesen.

```
public static void main(String[] args) {
    try {
        // select Look and Feel
        UIManager.setLookAndFeel(
            UIManager.getSystemLookAndFeelClassName());
    } catch (ClassNotFoundException | InstantiationException | IllegalAccessException | UnsupportedLookAndFeelException ex) {
        ex.printStackTrace();
    }
    view = new MainFrameView();
    model = new MainFrameModel(view.getFontModel(), view.getFontSizeMode(), view.getJTabbedPaneModel());
    controller = new MainFrameController(view, model);

    view.setVisible(true);
}
```

Abbildung 5: Syntaxhighlighting am Beispiel von Java

Auswahl der Programmiersprache

Beim Öffnen einer Datei versucht das Programm, anhand der Dateiendung (siehe Tabelle) die passende Programmiersprache zu erkennen. Gelingt dies nicht, kann die Sprache nachträglich über das Menü "Bearbeiten → Farben" eingestellt werden. Wird die Datei in einem Projekt gespeichert, wird auch die ausgewählte Programmiersprache gespeichert. Beim Laden des Projekts wird auch die Sprache wiederhergestellt.

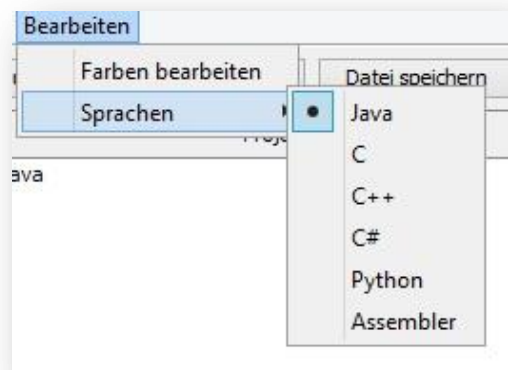


Abbildung 6: Menü zum Auswählen der Sprache

Automatische Erkennung der Programmiersprache

yate versucht anhand der Dateierdung zu erkennen, um welche Programmiersprache es sich bei dem enthaltenen Code handelt. Dabei werden nebenstehende Dateierdungen erkannt und den entsprechenden Programmiersprachen zugeordnet. Kann das Programm keine Sprache ermitteln, wird als Standardsprache Java ausgewählt.

Programmiersprache	Dateierdungen
C++	*.cpp, *.h, *.t
C	*.c
C#	*.cs
Java	*.java
Python	*.py
Assembler	*.asm

Farbeinstellungen

Über das Menü *"Bearbeiten → Farben bearbeiten"* lässt sich ein Dialog öffnen, in dem für die ausgewählte Sprache Farben für die einzelnen Elemente der Sprache eingestellt werden können. Die eingestellten Farben gelten für alle zur Laufzeit des Programms vorhandenen Dateien. Wird das Programm beendet, werden die Einstellungen verworfen, es sei denn sie werden innerhalb eines Projekts gespeichert.



Abbildung 7: Farbauswahldialog am Beispiel von Java

Am linken Rand des Dialogs werden die verfügbaren Bezeichner der ausgewählten Sprache aufgeführt. Diese variieren je nach Programmiersprache und sind an die individuellen Strukturen der Programmiersprache angepasst, wie nachfolgender Grafik zu entnehmen ist.

Schlüsselwort	Literal	Klammer	Literal
Bezeichner	Bezeichner	Bezeichner	Flag
Datentyp	Klammer	Schlüsselwort	Zahl
Literal	Datentyp	Kommentar	Memmonic
Klammer	Schlüsselwort	Literal	Register
Kommentar	Präprozessor		
	Kommentar		
Java	C/C++/C#	Python	Assembler

Abbildung 8: Sprachelemente der verfügbaren Programmiersprachen

Nachdem in der Liste am linken Rand ein Sprachelement ausgewählt wurde, kann über die Farbauswahl die gewünschte Farbe ausgewählt werden. Dabei stehen unterschiedliche Auswahlmodi für die Farben zur Verfügung.

Nachdem die gewünschten Einstellungen vorgenommen wurden, kann der Dialog mithilfe des Buttons "OK" verlassen werden. Die Farben in den geöffneten Dateien werden dadurch automatisch aktualisiert.

Syntaxeintrückung

Um den bearbeiteten Code besser zu strukturieren, bietet *yate* die Möglichkeit, Programmcode automatisch basierend auf der Klammersetzung einzurücken. Diese Funktion wird für die Sprachen C++, C, Java und C# unterstützt, da diese umfangreichen Gebrauch von geklammerten Codeblöcken machen.

yate bietet zwei Möglichkeiten, die automatische Syntaxeintrückung aufzurufen. Zum Einen wird die Einrückung automatisch durchgeführt, wenn eine schließende Klammer eingetippt wird. Allerdings kann es nötig sein (bspw. wenn Code eingefügt wurde), die Einrückung manuell aufzurufen. Dies kann durch die Tastenkombination "STRG+I" durchgeführt werden.

```

1 import java.util.*;
2
3 public class Main {
4     public static void main() {
5         System.out.println("Hello World");
6         for (int i=0; i<10; i++) {
7             System.out.println(i);
8         }
9     }
10 }
```

Abbildung 9: Vor der Einrückung

```

1 import java.util.*;
2
3 public class Main {
4     public static void main() {
5         System.out.println("Hello World");
6         for (int i=0; i<10; i++) {
7             System.out.println(i);
8         }
9     }
10 }
```

Abbildung 10: Nach der Einrückung

Klammerhervorhebung

Um in umfangreichen Programmen nicht den Überblick zu verlieren, welche Klammer zu welcher Programmstruktur gehört, bietet *yate* eine automatische Hervorhebung von Klammern an. Wenn der Cursor auf einer öffnenden oder schließenden Klammer positioniert wird, wird diese und ihr Gegenstück, sofern vorhanden, farblich hervorgehoben.

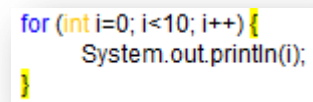


Abbildung 10:
Klammerhervorhebung

Projektverwaltung

Um verschiedene Dateien zu gruppieren, bietet *yate* die Möglichkeit, diese in Projekten zusammenfassen. Projekte sind spezielle Dateien, die Informationen über die in ihnen enthaltenen Textdateien, einen Projektnamen, Spracheinstellungen für die Dateien und Farbeinstellungen enthalten.

Das Projekt, das zurzeit geöffnet ist, wird am linken Rand der Oberfläche in einem Projektmenü angezeigt.

Datei hinzufügen

Durch einen Klick auf den Button "+" wird die aktuell fokussierte Datei dem Projekt hinzugefügt, sofern sie noch nicht Teil des Projekts ist. Sie erscheint daraufhin in der Liste der Dateien.

Datei entfernen

Durch einen Klick auf den Button "-" wird die Datei, die in der Liste ausgewählt, aus dem Projekt entfernt.

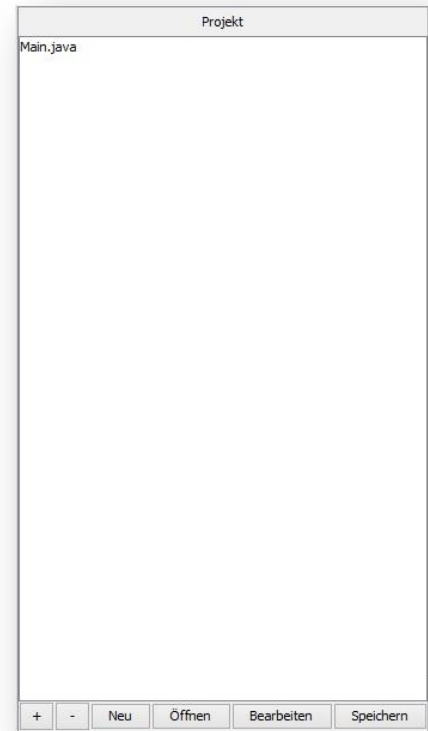


Abbildung 11: Projektmenü

Öffnen

Über diesen Button kann ein zuvor gespeichertes Projekt geladen werden. Dadurch gehen Änderungen an dem aktuellen Projekt verloren.

Bearbeiten

Über diesen Button kann der Name des aktuellen Projekts bearbeitet werden. Der geänderte Name wird am oberen Rand des Projektmenüs angezeigt.

Speichern

Der "Speichern"-Button speichert den aktuellen Zustand des Projekts. Wenn für das Projekt noch keine Dateiinformationen, wie der Speicherort und der Name der Datei, hinterlegt worden sind, werden diese über einen Dialog abgefragt. Andernfalls werden die hinterlegten Speicherinformationen verwendet. Beim Speichern des Projekts werden der Name, die Liste der Dateien, die eingestellten Sprachen, sowie die konfigurierten Farbeinstellungen gespeichert.

Autovervollständigung

Bei der Programmierung kommt es häufig vor, dass lange Bezeichner oder Schlüsselwörter genutzt werden. Da sich diese häufig wiederholen bietet *yate* eine automatische Vervollständigung von bereits eingegebenen Wörtern an. Während ein Wort eingegeben wird, durchsucht *yate* automatisch das Dokument nach Begriffen, deren Anfang mit dem bereits eingegebenen Wortteil übereinstimmt. Wird ein Treffer gefunden, wird dieser vorgeblendet und kann durch Drücken der "TAB"-Taste eingefügt werden. Werden mehrere Treffer gefunden, kann die Liste der Treffer mithilfe von "ALT + Pfeil hoch" bzw. "ALT + Pfeil runter" durchlaufen werden.

```
public static void main() {  
    int langerBezeichner = 0;  
    int langerBezeichner2 = 0;  
    langerBezeichner  
}
```

Abbildung 12:
Autovervollständigung