

Protokoll #10: WordStat

Dieses Dokument enthält folgende Themen:

1. Ziel
2. Ausstattung der Hard-/Software
3. Programmliste
4. Funktionsbeschreibung
5. Programmtest
6. Benutzeranleitung
7. Erkenntnisgewinn

Ziel

Das Ziel und die Aufgabe war es, eine SDI¹-Anwendung für eine beliebige Aufgabe zu erstellen, die graphische Ausgaben tätigt.

Ausstattung der Hard-/Software

Erstellt und getestet wurde das C++-Programm WORDSTAT auf meinen heimischen PC:

- Prozessor: AMD-Athlon mit 1 GHz und 512 MB
- Betriebssystem: Windows XP SP2 (5.1 Version 2002 Service Pack 2)

Als Entwicklungsumgebung wurde folgende IDE & Compiler verwendet:

- **Visual C++ 6.0 Enterprise-Edition** © Microsoft Corporation

¹SDI steht für *Single Document Interface*.

Programmliste

Das gesamte Werk besteht aus diesem Protokoll, sowie einem Archiv mit allen C++-Quelldateien & Ressourcen mitsamt einem ausführbaren Kompilat:

- Protokoll: Michael Johné #10 PP_A WordStat .doc | .odt | .pdf
- C++-Quelltexte: Michael Johné #10 PP_A WordStat.zip (Dieses Archiv enthält alle benötigten Quelltexte, Ressourcen und sonstige Dateien).
- Kompilat: Michael Johné #10 PP_A WordStat.exe
- Referenztext: genesis-de.txt
- Weitere Referenztexte („xx“ steht für die jeweilige Sprache): genesis-xx.txt
- Vergleichsobjekt als MS Excel-Tabelle: genesis-de.xls

Das Protokoll beschreibt in erster Linie die **Zielsetzung**, die **Benutzungs-/Funktionweise** und den **Erkenntnisgewinn** der erstellten C++-Quelldatei.

Weiterhin sind Beschreibungen & Interpretationen zu den unterschiedlichen Diagramme enthalten, die mit WORDSTAT erzeugt werden können.

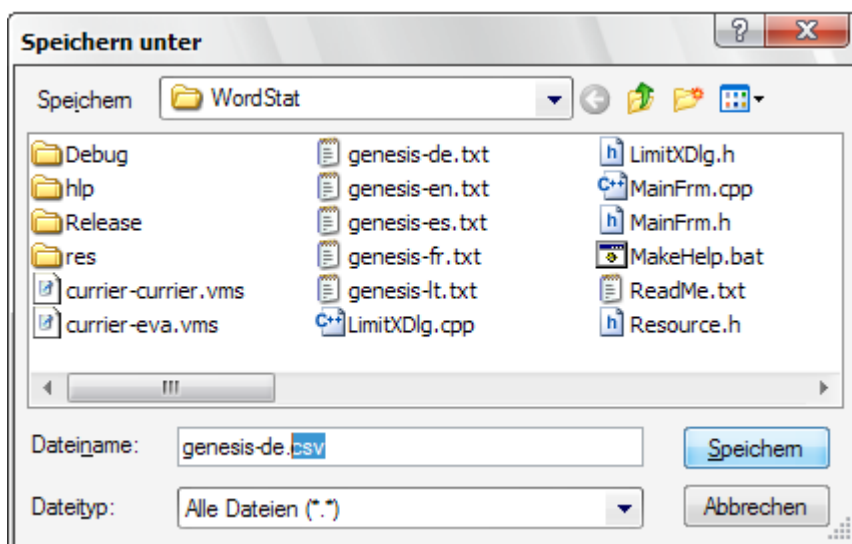
Funktionsbeschreibung

Was leistet das C++-Programm WORDSTAT? Dieses Programm liest den Inhalt einer Textdatei und ermittelt die darin enthaltenden Wörter, sowie ihre Häufigkeiten und Wortlängen. Die Wörter werden in einer Worttabelle abgespeichert und können als CSV-Datei exportiert werden. Weiterhin besteht die Möglichkeit, die Worttabelle in mehrere unterschiedliche Diagrammen graphisch darzustellen.

Programmtest

Das C++-Programm WORDSTAT wurde nach Erstellung & Kompilierung mit *Microsoft Visual C++ 6.0* getestet und ausgeführt. Sämtliche verwendete Funktionen verhalten sich wie geplant.

Doch es kann zu Problemen bei der Speicherung ins CSV-Format kommen. Standardmäßig wird im Dateidialog, der beim Speichern erscheint, der Dateiname der Eingabedatei angezeigt. In der Regel ist die Dateiendung immer `TEXT`. Für das CSV-Format benötigt man allerdings die Dateiendung `CSV`. Das heißt, dass der Benutzer genau aufpassen muss, dass er die Dateiendung `CSV` nach dem Dateinamen eingibt:

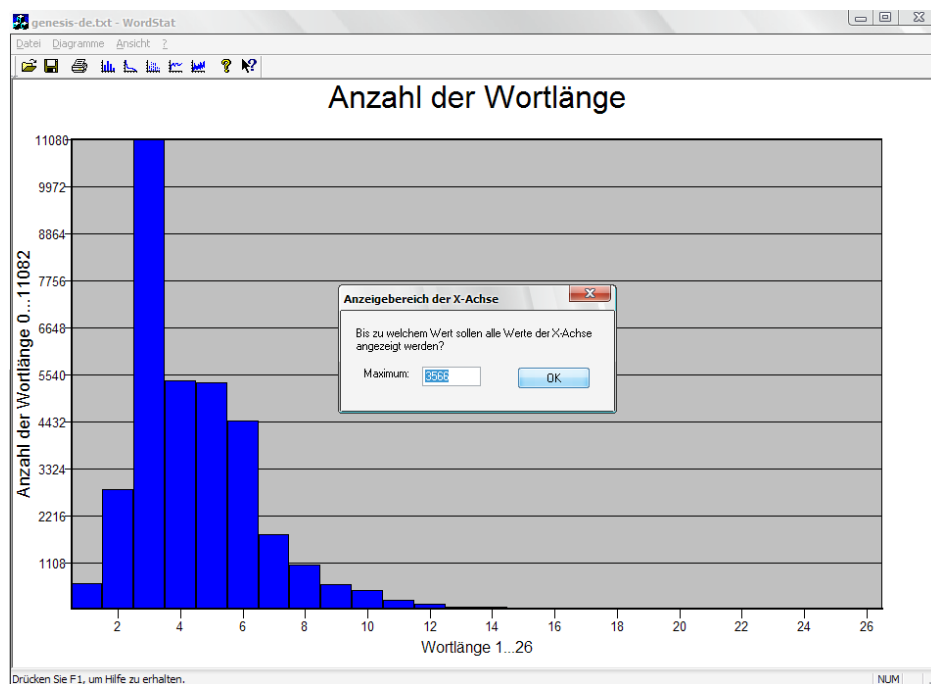


Das zweite Problem betrifft den Algorithmus, der sämtliche Wörter aus einem Text ermittelt und in eine Worttabelle einträgt. Dieser Algorithmus ist noch sehr langsam und führt fälschlicherweise dem Benutzer zum Verdacht, das Problem habe sich aufgehängt. Der Algorithmus liest dabei Zeile für Zeile einer Textdatei ein und betrachtet die Zeile als ein Feld von Zeichen. Gerade hier wird beim Bilden von Wörtern und mit dem Vergleichen aus der Worttabelle viel Zeit benötigt.

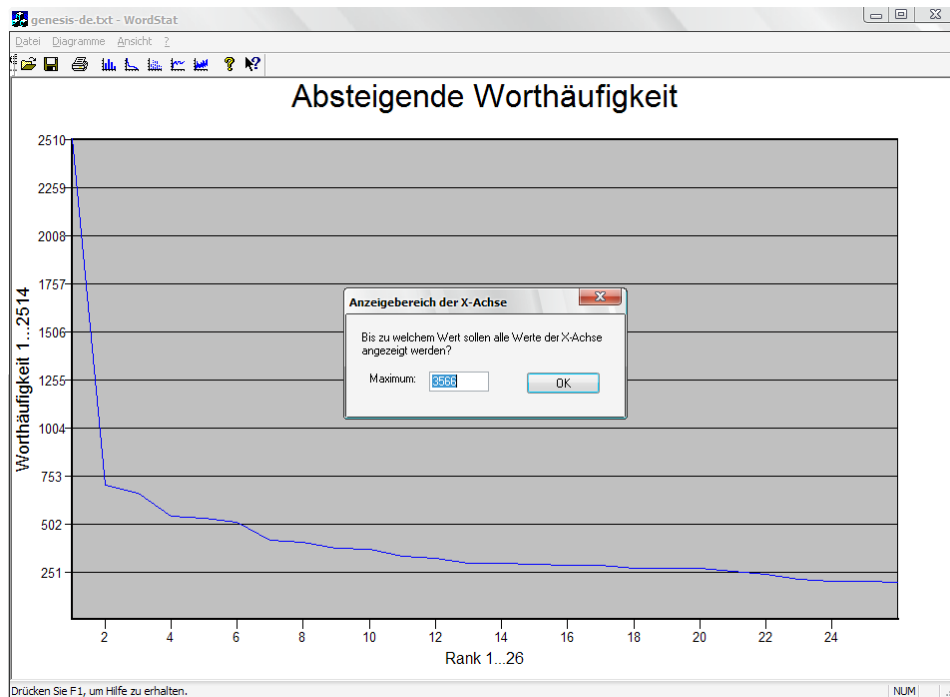
(Ich denke, wenn man dem Gedanken bzw. der Benutzung eines Feldes von Zeichen verwirft und stattdessen sich auf „Zeichenketten-Basteleien“ der CString-

Klasse der MFC einlässt, könnte man hier bis zu ein Drittel Zeit und Geschwindigkeit gewinnen.)

Ein weiteres Problem, das entdeckt worden war, betrifft das Zeichnen von Diagrammen im Hintergrund. Nachdem man bereits ein Diagramm zum Anzeigen ausgewählt hat und nun ein neues Diagramm anzeigen möchte, muss man zuerst den Anzeigebereich der X-Achse auswählen:

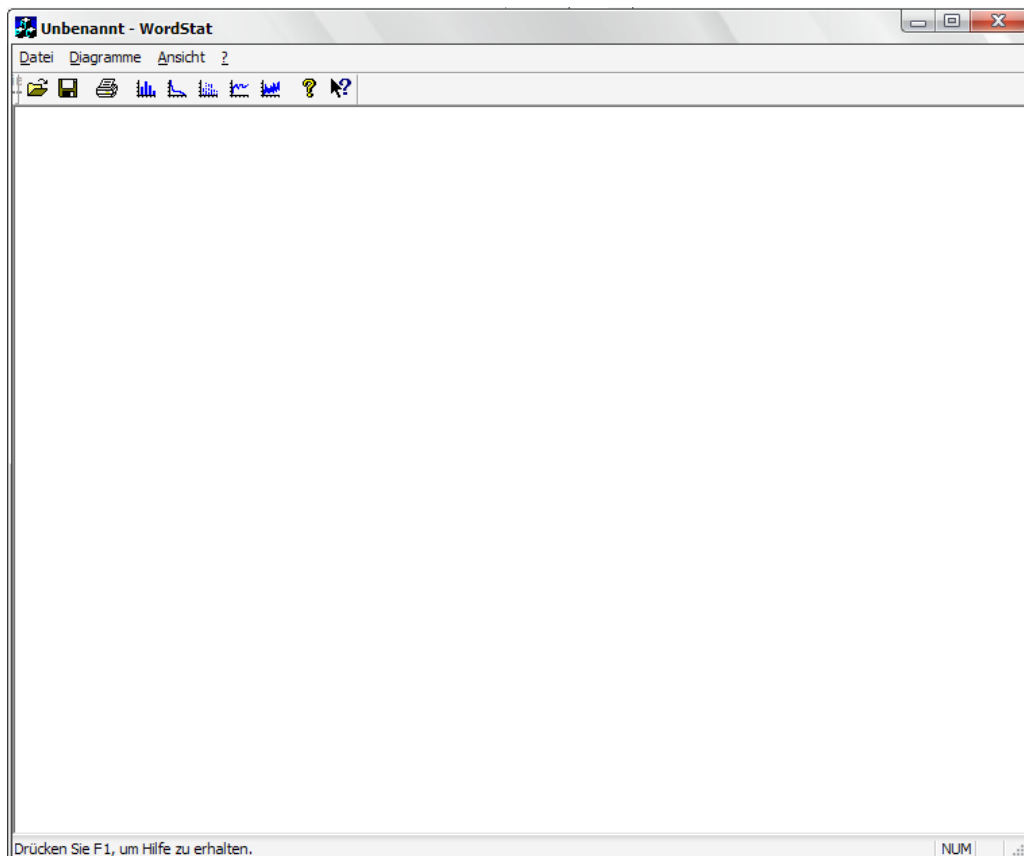



Noch während man den Wert im erscheinenden Dialog eingibt, kurzzeitig ein anderes Programm aufruft und dann wieder WORDSTAT, hat es zur Folge, dass im Hintergrund schon das neue Diagramm gezeichnet, noch bevor der Dialog zur Anzeigebegrenzung der X-Achse geschlossen worden ist.

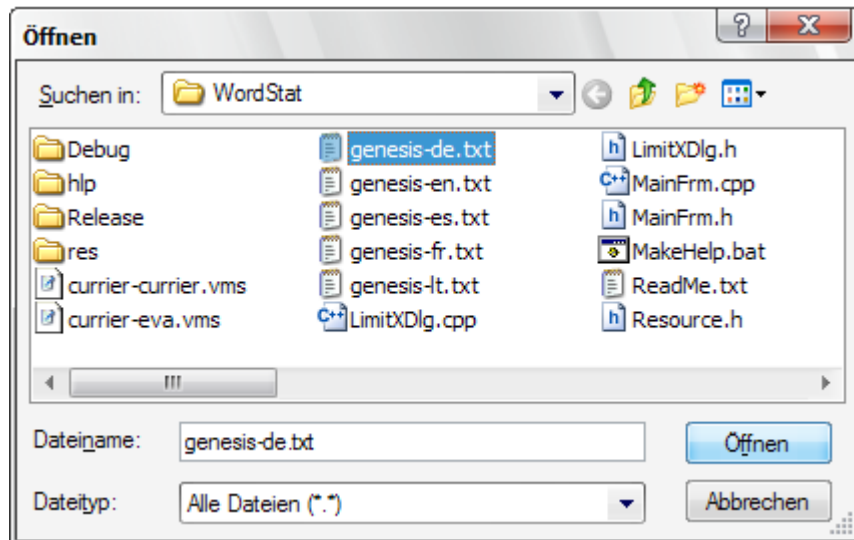


Benutzeranleitung

Zu Beginn zeigt das Programm WORDSTAT eine leere Programmoberfläche:

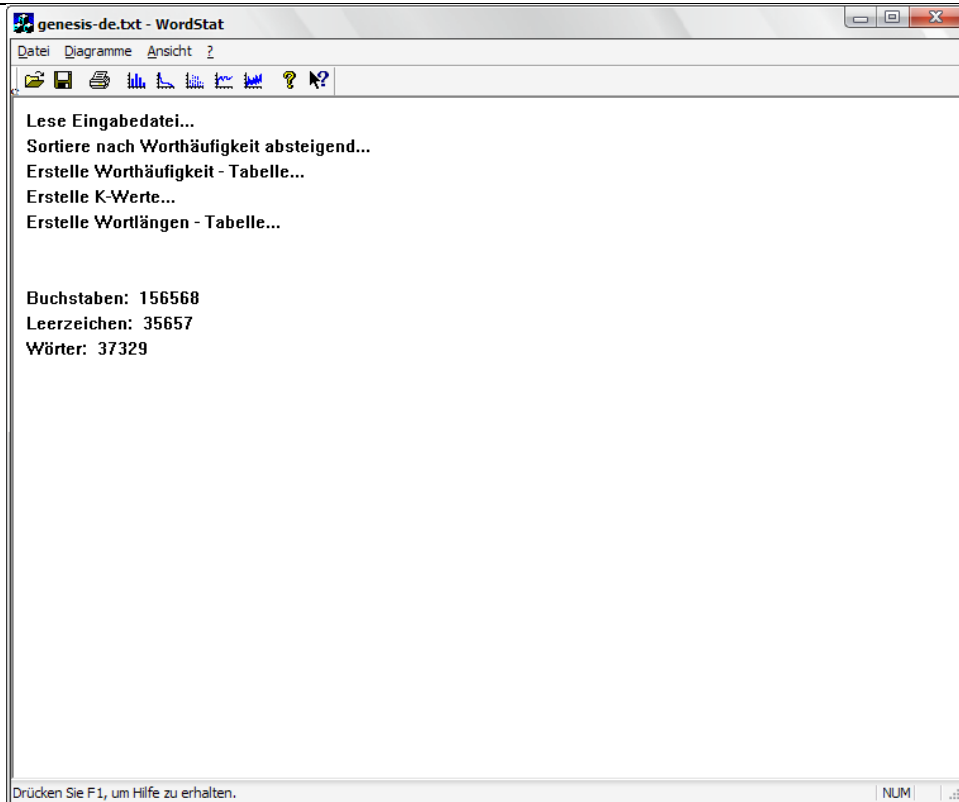


Auf das Ordnersymbol  oder alternativ im Menüeintrag „Datei > Öffnen“ kann man eine Textdatei auswählen, die geöffnet und analysiert werden soll:



Im weiteren Verlauf der Benutzeranleitung benutzen wir in der obigen Abbildung die markierte Textdatei `genesis-de.txt`. Sie beinhaltet das komplette erste Buch *Genesis* des AT aus der Bibel.

Nach dem Auswählen und Öffnen wird die Textdatei analysiert. Allerdings dauert einige Sekunden, bis der Vorgang komplett abgeschlossen ist. Im Anzeigefenster, dass danach erscheint, dann man erkennen, welche Vorgänge getätigt worden sind:

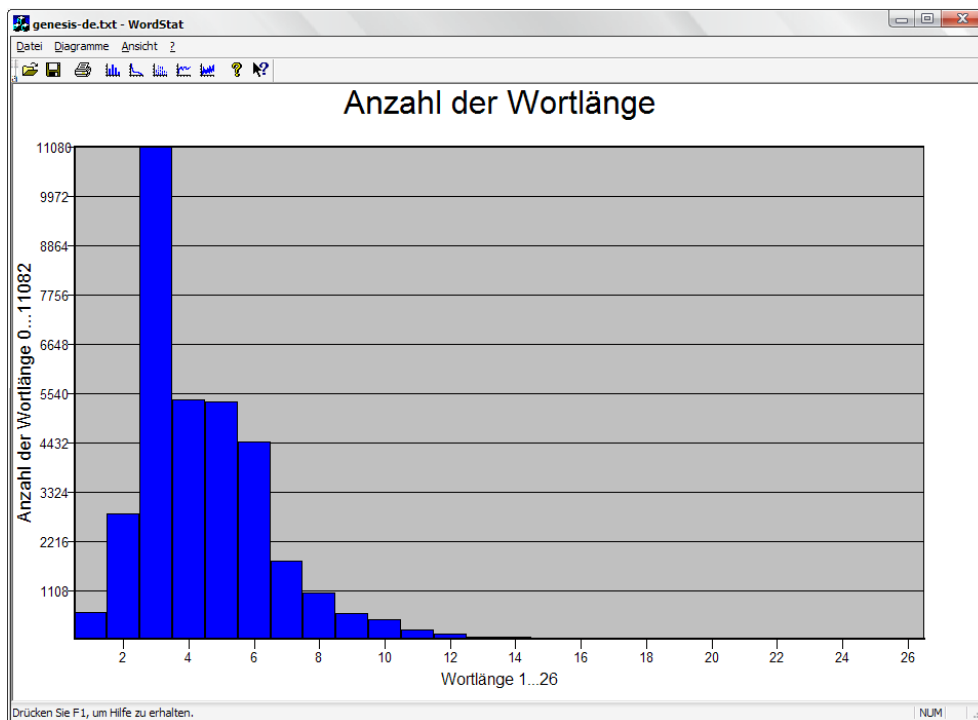


Nun stehen verschiedene Arten von Diagrammen zur Anzeige zur Verfügung:

- Wortlängen – Diagramm (auch *Anzahl der Wortlängen* genannt)
- Worthäufigkeit – Diagramm (auch *Absteigende Worthäufigkeit* genannt)
- Wortlängen-\-häufigkeit – Diagramm (auch *Relation zwischen Wortlänge und Worthäufigkeit* genannt)
- K-Wert – Diagramm
- Folge von Wortlängen

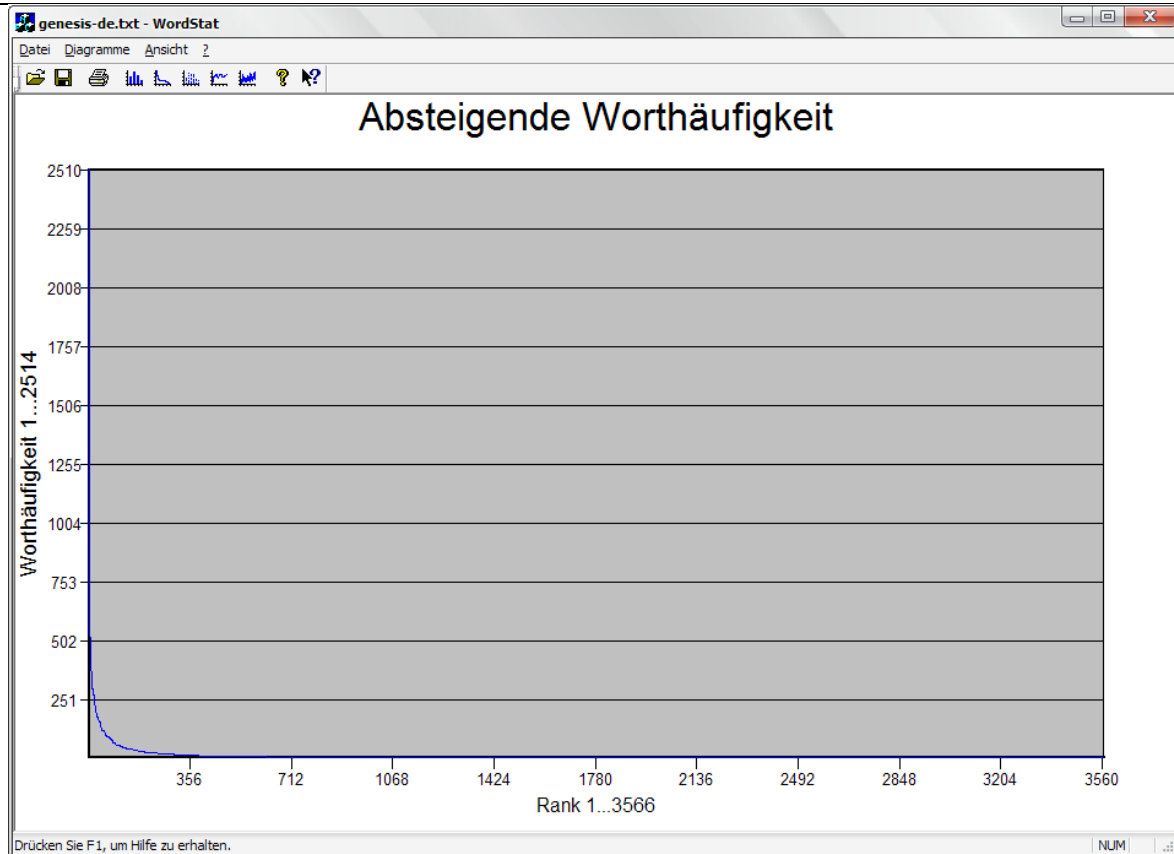
Wortlängen - Diagramm

Das Wortlängen – Diagramm definiert sich wie folgt: Es stellt die Anzahl aller Wörter je Wortlänge in einem Säulendiagramm dar. Aus dem Wortlängen – Diagramm ist zu erkennen, bei welchen Wortlängen die meisten bzw. wenigsten Wörter vorzufinden sind. In der folgenden Abbildung kann man zum Beispiel erkennen, dass bei der Wortlänge 3 die meisten Wörter vorzufinden sind. Ab Wortlängen von 10 Buchstaben sind hingegen sehr wenige Wörter vorhanden.

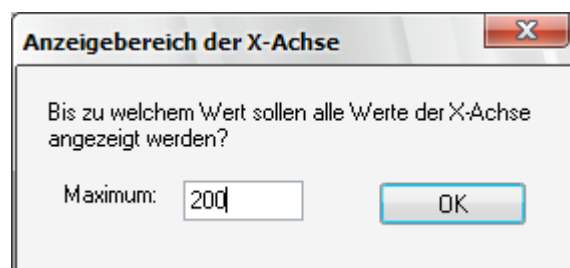


Worthäufigkeit - Diagramm

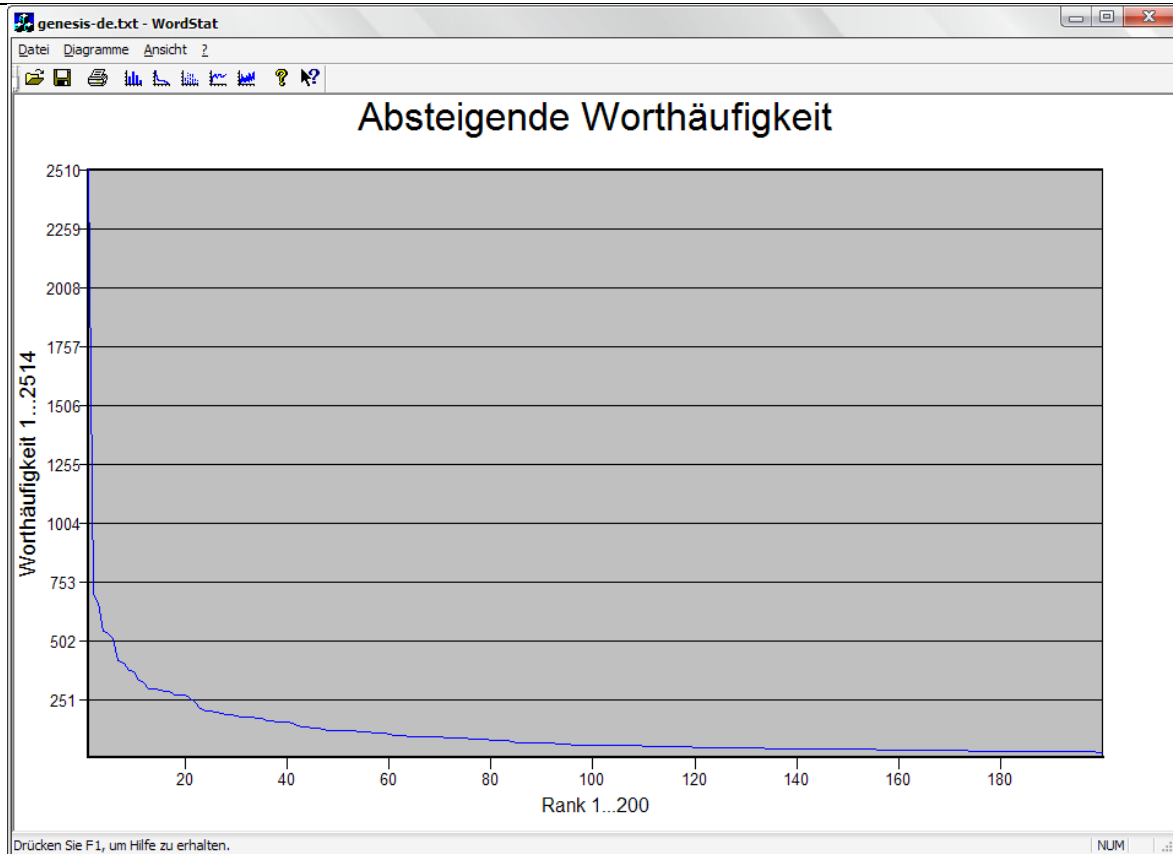
Das Worthäufigkeit – Diagramm definiert sich wie folgt: Es stellt eine absteigende Häufigkeit der vorhandenen Wörter in einem Liniendiagramm dar, d. h. die Worthäufigkeit sämtlicher Wörter wird absteigend sortiert. Die Plätze der Wörter nach absteigender Sortierung werden als Ränge bezeichnet. Aus dem Worthäufigkeit – Diagramm ist zu erkennen, welche Wörter welche Häufigkeit besitzen und wie sich die Gesamtheit der Häufigkeiten absteigend sortiert verhält, z. B. linearer oder exponentieller Abstieg. In den folgenden Abbildungen kann man zum Beispiel erkennen, dass bei den ersten 300 Wörtern eine erkennbare Größe herrscht, während bei den restlichen Wörtern eine minimale Häufigkeit herrscht.



Manchmal ist es auch wichtig, die Verteilung der Worthäufigkeit nur der ersten Wörter zu kennen. Hier begrenzt man den Anzeigebereich der X-Achse auf ein geringes Mindestmaß, der zu Beginn in einem Eingabefeld eines erscheinenden Dialogs einzugeben ist:



Danach wird das Worthäufigkeit – Diagramm mit der gewünschten Anzeigegrenzung der X-Achse angezeigt:



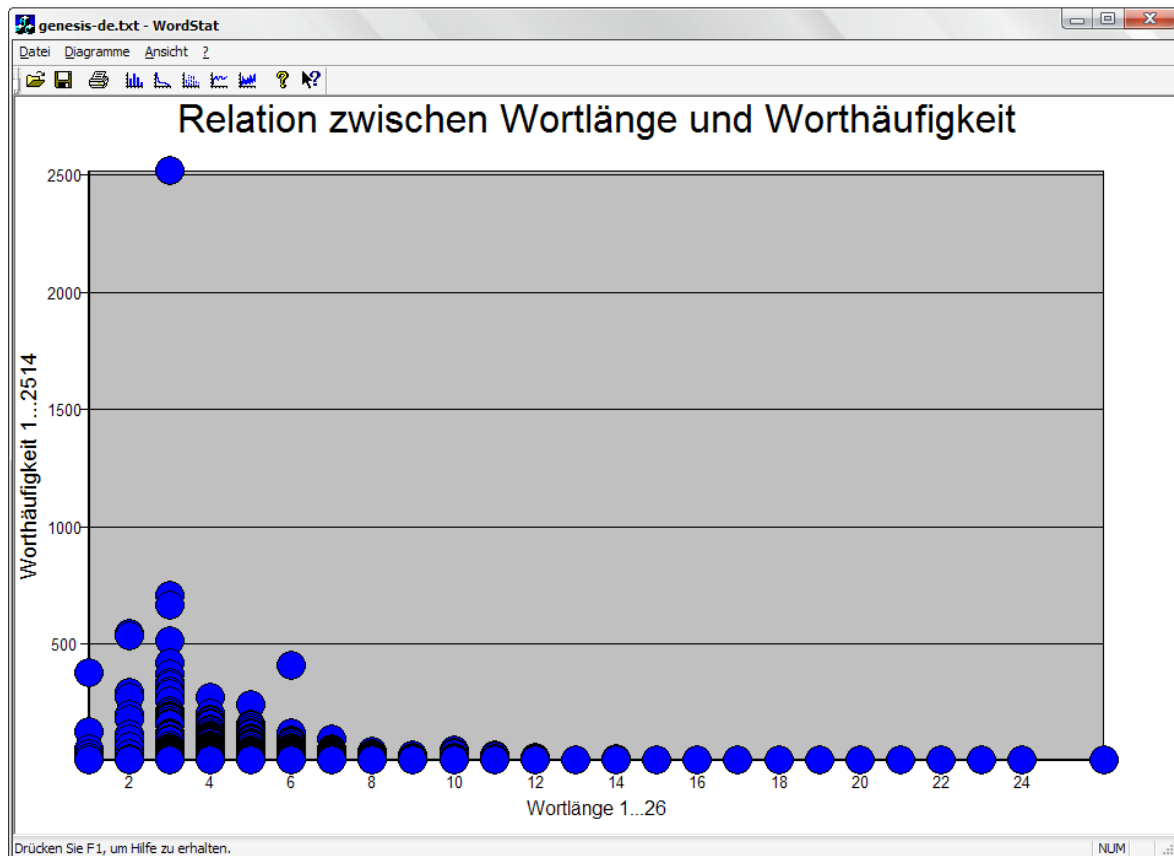
In Texten, die sprachlichen Konstruktionsregeln unterliegen, besitzt der Graph immer eine exponentielle Form (ähnlich einer Hyperbel $y(x) = x^{-a}$). Diese Regel ist durch das so genannte *Zipf-Gesetz* beschrieben. Dieses Gesetz wurde in den 30-iger Jahren von George Kingsley Zipf entdeckt. Es sagt aus, dass bei bestimmten Größen zu dem Rang x eines Wortes e mit einem gesuchten Potenzwert a der zugehörige Wert y sich abschätzen lässt.

Wortlängen-\-häufigkeit – Diagramm

Das Wortlängen-\-häufigkeit – Diagramm definiert sich wie folgt: Länge und Häufigkeit eines Wortes werden als X,Y-Koordinaten eines Punktes verstanden und in ein Punktdiagramm eingetragen. Aus dem Wortlängen-\-häufigkeit – Diagramm ist dann zu erkennen, welches Wörter welche Häufigkeit mit einer bestimmten Länge hat. Besonders gut und schnell lassen sich dann auch sehr häufig auftretende oder seltene Wörter erkennen. In der Regel sind die häufigsten Wörter Binde- wörter, Pronomen oder Artikel. So lassen sich auch in einem monoalphabetisch

verschlüsselten Text mögliche Bindewörter, Pronomen oder Artikel abschätzen oder identifizieren, wenn man nicht die Sprache kennt oder eine Zeichenhäufigkeitsanalyse fehlschlägt, z. B. bei einer Plansprache.

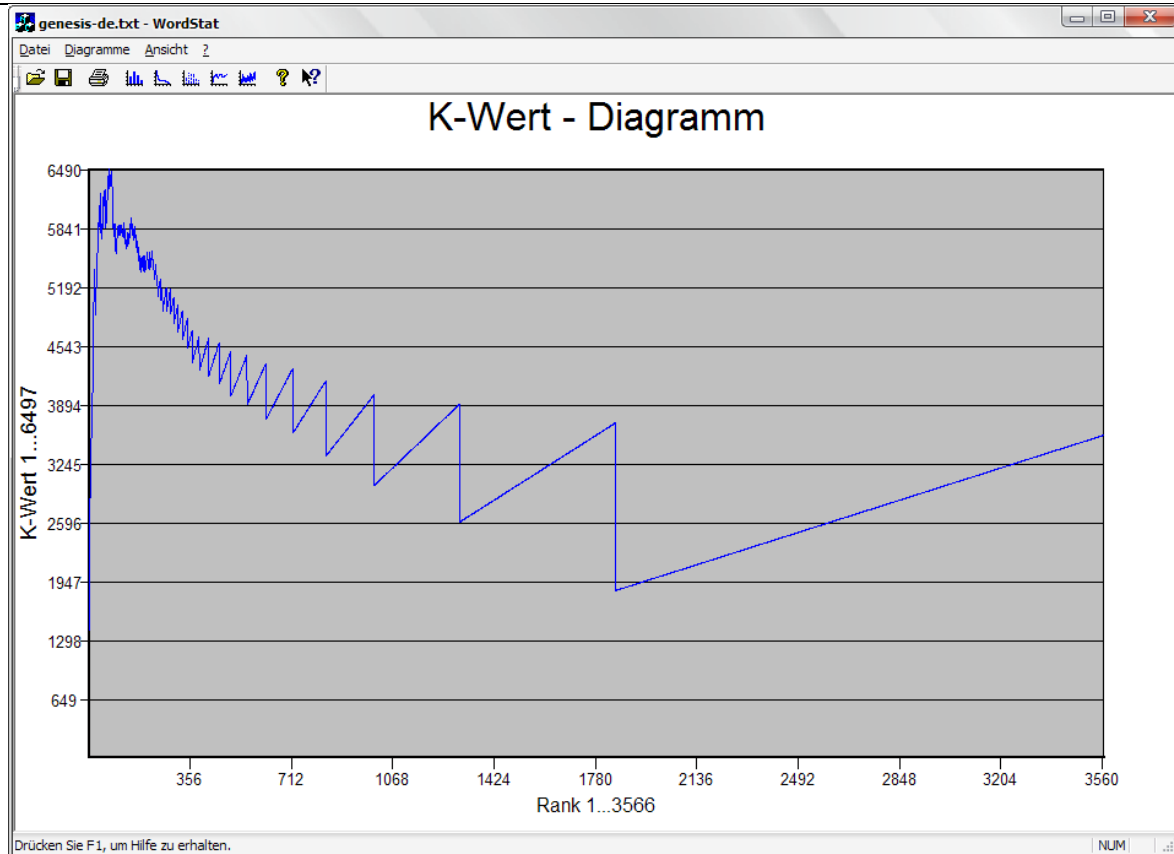
In der folgenden Abbildung entspricht z. B. der höchste Punkt dem deutschen Bindewort *und*:



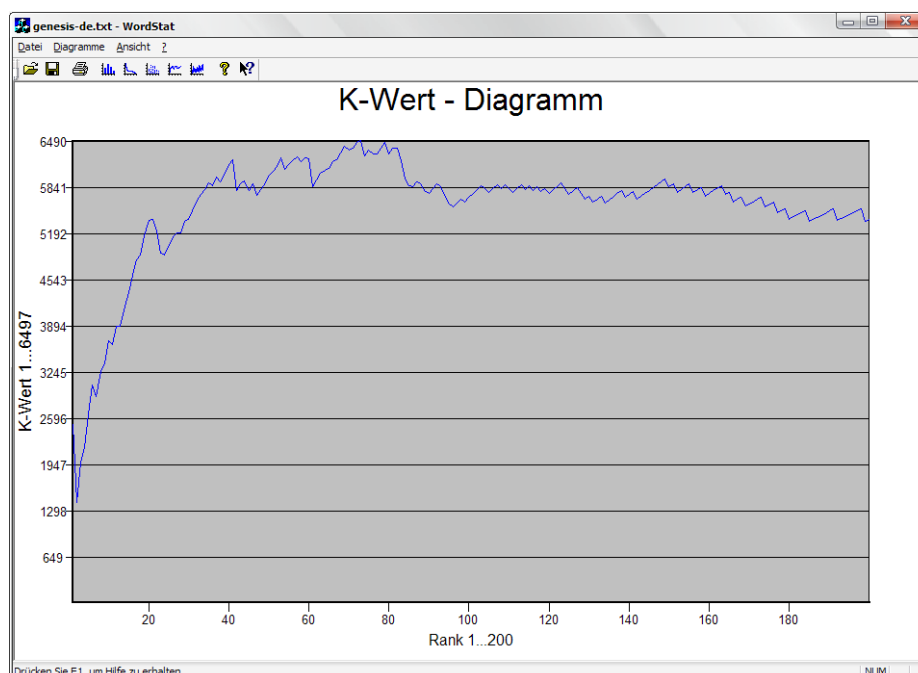
K-Wert – Diagramm

Das K-Wert – Diagramm definiert sich wie folgt: Multipliziert man den Rang eines Wortes mit seiner Häufigkeit, dann erhält man den K-Wert eines Wortes. Der Graph sämtlicher K-Werte muss in der Theorie eines konstanten Linien entsprechen. In der Praxis wird sich jedoch nie geschehen²:

²Leider kann ich nicht mitteilen, wozu der K-Wert gut ist bzw. welchen Zweck er erfüllt und warum sein Graph in der Theorie konstant ist, da einige Informationen nicht zur Verfügung standen: Ich weiß nur, dass es ihn gibt, dass es angewandt wird und dass er in der Theorie konstant ist.

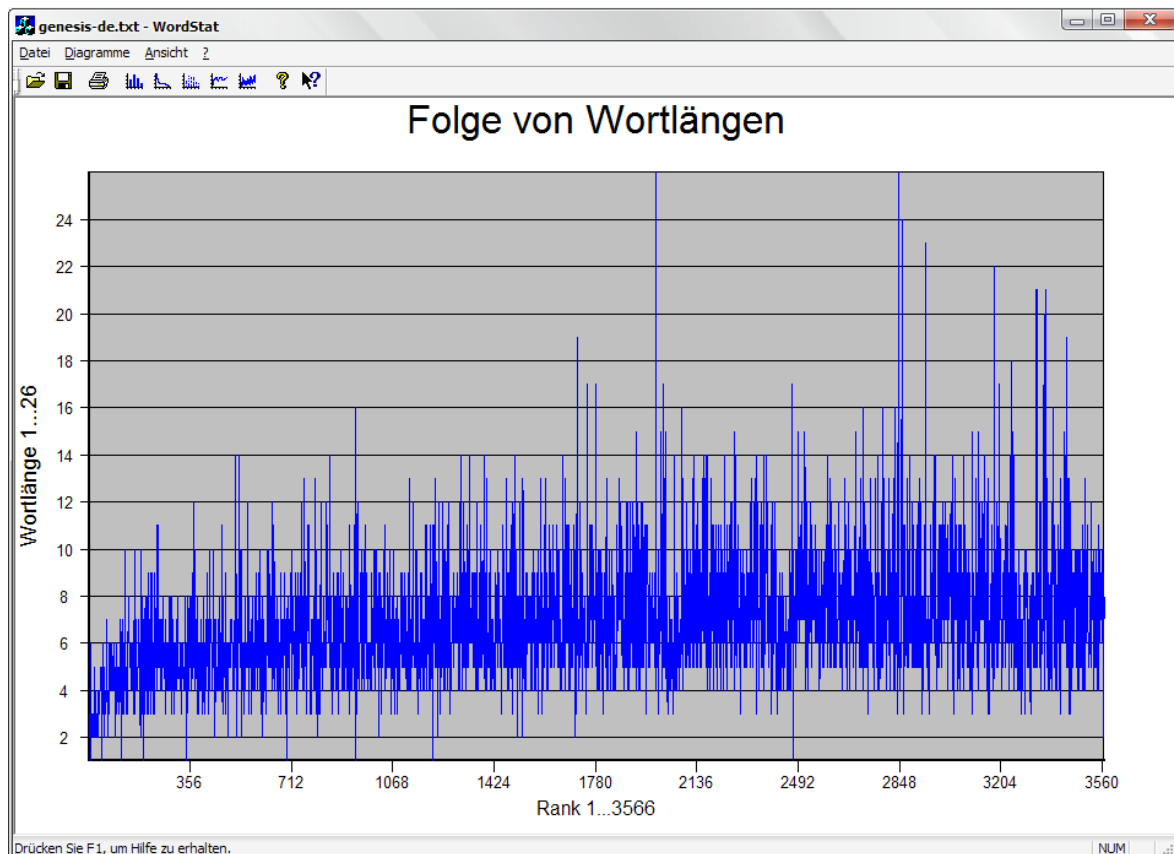


Unter Umständen sind auch nur die ersten K-Werte wichtig, da die meisten K-Werte je Wortlänge linear verlaufen. Hierzu kann man wieder den Anzeigebereich der X-Achse begrenzen (z. B. auf 200) und man erhält folgende Anzeige:



Folge von Wortlängen

Das Diagramm mit den Folgen von Wortlängen definiert sich wie folgt: Es stellt die Längen sämtlicher Wörter einer absteigenden Sortierung in einem Liniendiagramm dar. Aus dem Diagramm mit den Folgen von Wortlängen ist zu erkennen, wie sich die Wortlängen von den häufigsten Wörtern bis zu den seltensten Wörtern verhält und welches Mittelmaß abgeleitet werden kann. Nach den Gesetzen der Linguistik ist es, dass die häufigsten Wörter sehr kurze Wortlängen enthalten, während seltene Wörter längere Wortlängen beinhalten. Mit dem Diagramm der Folgen von Wortlängen soll dies bewiesen und erkannt werden, welcher Verlauf angenommen wird:



Aus dieser Abbildung erkennt man, dass zu Beginn der ersten hundertsten häufigsten Wörter die Wortlängen im Durchschnitt unter 8 Buchstaben liegt und in den weiterem Verlauf der Durchschnitt stagnierender wird.

Erkenntnisgewinn

Die Diagramme, die in WORDSTAT verwendet werden, basieren auf den Vorlagen aus dem letzten Projekt SDI_TEST. Für WORDSTAT wurden nun die Diagramm sowohl im optischen Design als auch in ihrem Verhalten von den Eigenschaften der Diagramm von *MS Excel* angepasst. Bewusst nicht berücksichtigt wurde dabei eine minimale Fenstergröße der WORDSTAT-Anwendung.

Etwas umständlich ist zudem auch noch die Maßeinteilung bei den Achsen und die Verwaltung des Quellcodes. Bei den Maßeinteilung tauchen i. d. R. immer „krumme“ Zahlen auf, d. h. Maße, die nicht auf 0, 5, oder geraden Zahlen enden. Der Grund ist hierbei, dass die Maßeinteilung vom Maximalwert in 10 Stücke geteilt wird.

Bei der Verwaltung des Codes wird für jedes Diagramm der dazu nötige Quellcode geschrieben, d. h. es wird nicht auf eine mögliche (Template-)Klasse zurück gegriffen.

Mit dieser Projektarbeit wurde auch eine wichtige Methode von SDI-basierten Programmen kennen gelernt: die Methode *UpdateAllViews()*. Mit dieser Methode können Informationen auf der Zeichenoberfläche geschrieben werden, noch während man Bearbeitungsvorgänge tätigt.

Zudem wurde die Erkenntnis gewonnen, wie man Dialoge in SDI-basierten Anwendung implementiert und einsetzt.