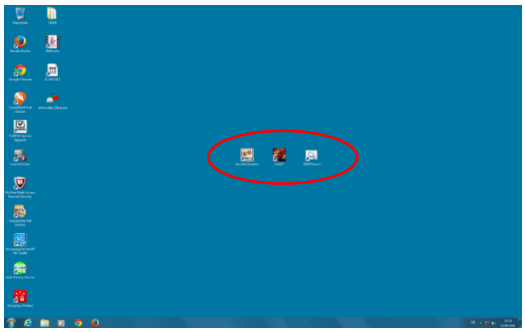
1 Suchanfragetools

Im Kiezdeutschkorpus können mithilfe der drei unterschiedlichen Tools ANNIS, EXAKT (Exmaralda) und TIGERSearch verschiedene Suchanfragemethoden genutzt werden. Diese Tools können durch Anklicken der entsprechenden Symbole auf dem Desktop geöffnet werden.



ANNIS[1] ist ein browser-basiertes Suchtool für annotierte Korpora. Es wurde ursprünglich im Rahmen des Sonderforschungsbereichs 632 „Informationsstruktur: Die sprachlichen Mittel der Gliederung von Äußerung, Satz und Text“ an der Humboldt-Universität zu Berlin und der Georgetown University entwickelt und unterstützt eine Vielzahl an Annotationen.
Durch *ANNIS* können im Kiezdeutschkorpus Suchanfragen nach syntaktischen Mustern oder einfachen Lexemen gestellt werden. Außerdem ist die Suche auf unterschiedlichen Sprecher-, Annotations-und Normalisierungsebenen möglich. Das Programm gibt bei Bedarf den Ausschnitt der Audiodatei der entsprechenden Funde wider und beinhaltet zudem Metainformationen zu den Sprecherinnen und Sprechern. ANNIS kann online und offline genutzt werden. Aus Datenschutzgründen können in der online-Version keine Audiodaten dargestellt werden; dies ist aber in einer offline-Version möglich, auf die an einem lokalen KiDKo-Arbeitsplatz in Potsdam zugegriffen werden kann. Hierfür kann man jederzeit Slots zur Arbeit an KiDKo reservieren. Mittelfristig wird eine Suche der syntaktisch annotierten Version des Korpus möglich sein, sodass die Suche von topologischen Feldern in Baumstrukturen, ähnlich wie bei TIGERSearch, möglich ist.

http://www.http://corpus-tools.org/annis/ [2]

TIGERSearch ist ein java-basiertes Suchtool für syntaktisch annotierte Korpora, das von 1993 bis 2003 am Institut für Maschinelle Sprachverarbeitung an der Universität Stuttgart von Dr. Wolfgang Lezius entwickelt wurde.
Mit *TIGERSearch* kann im Kiezdeutschkorpus auf die morphologisch und syntaktisch annotierte Version des Korpus zugegriffen werden. Gesucht werden kann entweder mithilfe des Textfeldes und der TIGERSearch Query Language oder über die graphische Oberfläche. Es können Lexeme, Wortarten und syntaktische Feldkategorien gesucht werden. Das Programm gibt Strukturbäume aus, die auf dem topologischen Feldermodell beruhen. Es können außerdem gezielt bestimmte Relationen verschiedener Elemente oder syntaktische Strukturen im

[1] Krause, Thomas & Zeldes, Amir (2014): *ANNIS3: A new architecture for generic corpus query and visualization.* in: Digital Scholarship in the Humanities 2014
http://dsh.oxfordjournals.org/cgi/content/abstract/fqu057?ijkey=GJBr0LhNfKW1g8i&keytype=ref

[2] Die Website bietet auch eine Übersicht über alle aktuellen und ehemaligen Mitarbeiter/innen.

Strukturbaum gesucht werden. TIGERSearch ermöglicht außerdem automatische Frequenzanalysen. Im Gegensatz zu ANNIS wird TIGERSearch ausschließlich lokal genutzt.

http://www.ims.uni-stuttgart.de/forschung/ressourcen/werkzeuge/tigersearch.html

EXAKT ist das Such- und Analysewerkzeug für annotierte Korpora in EXMARaLDA. EXMARaLDA entstand im Sonderforschungsbereich „Mehrsprachigkeit“ (SFB 538) an der Universität Hamburg und wird seit 2011 durch das Hamburger Zentrum für Sprachkorpora in Zusammenarbeit mit dem Archiv für Gesprochenes Deutsch des IDS Mannheim weitergeführt. Das KiezDeutsch-Korpus lässt sich mit EXAKT unter Verwendung von regulären Ausdrücken durchsuchen. Zudem ist es möglich, sich den Kontext zum Fund anzeigen und die entsprechende Audiodatei abspielen zu lassen. Suchergebnisse können automatisch oder manuell gefiltert und sortiert werden. Außerdem können Metadaten zu den Sprecher/inne/n abgerufen werden. EXAKT wird wie TIGERSearch lokal benutzt und steht ebenfalls auf dem lokalen KiDKo-Arbeitsplatz in Potsdam zur Verfügung.

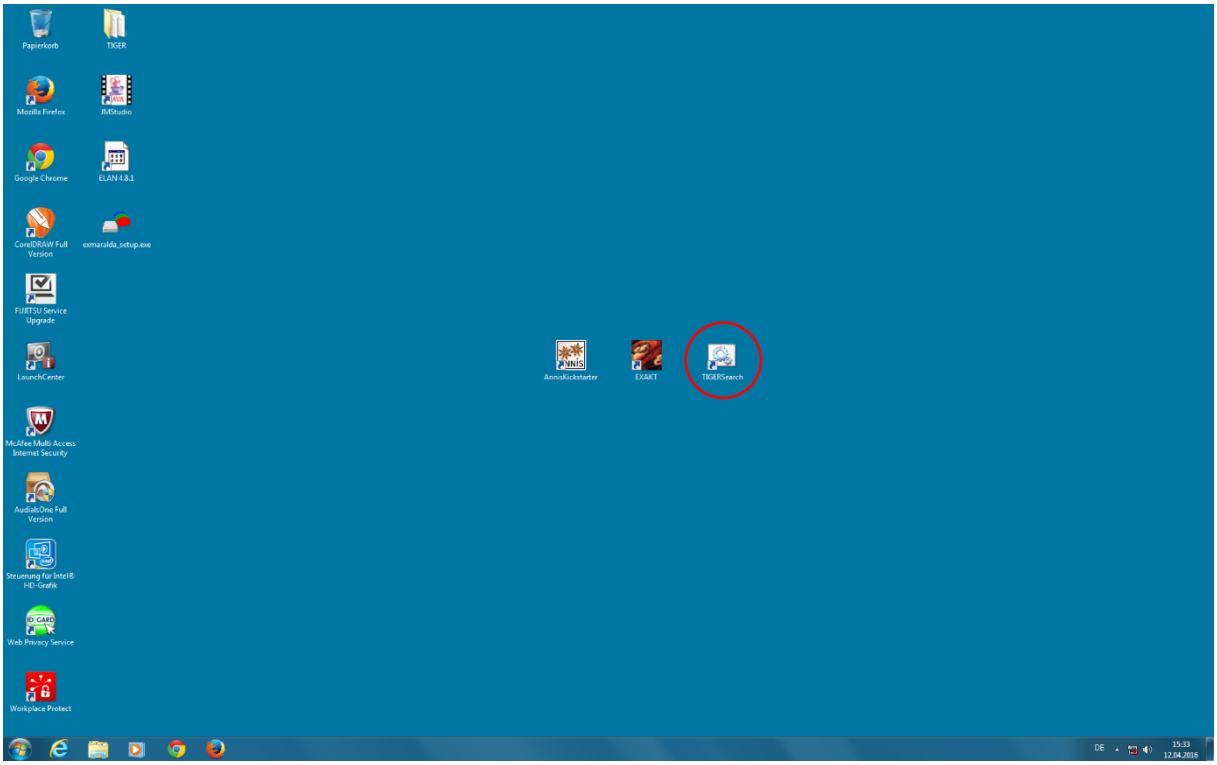
http://www.exmaralda.org/tool/exakt/

Inhalt

1.1 TIGERSearch

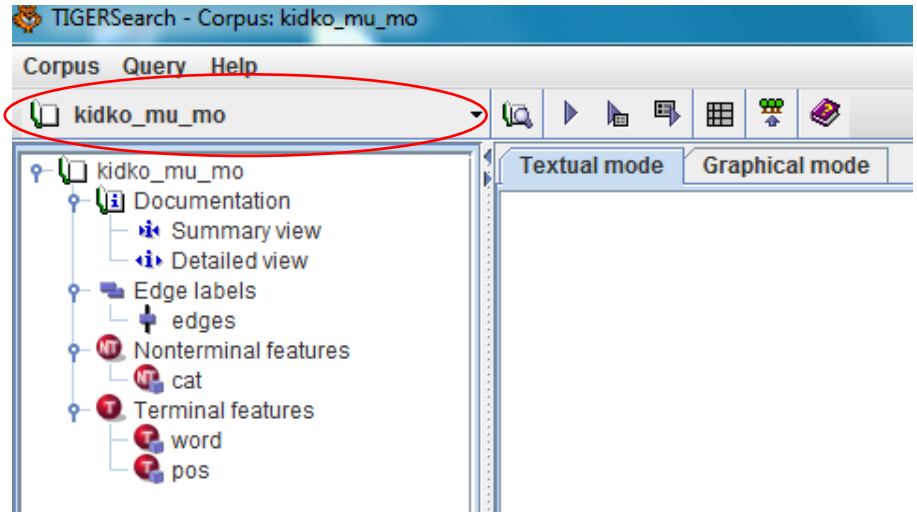
1.1.1 TIGERSearch starten

TIGERSearch wird per Klick auf das Symbol auf dem Desktop gestartet.



1.1.2 Korpus öffnen

Das Korpus kann durch die Auswahl links oben im Programmfenster gewählt werden.



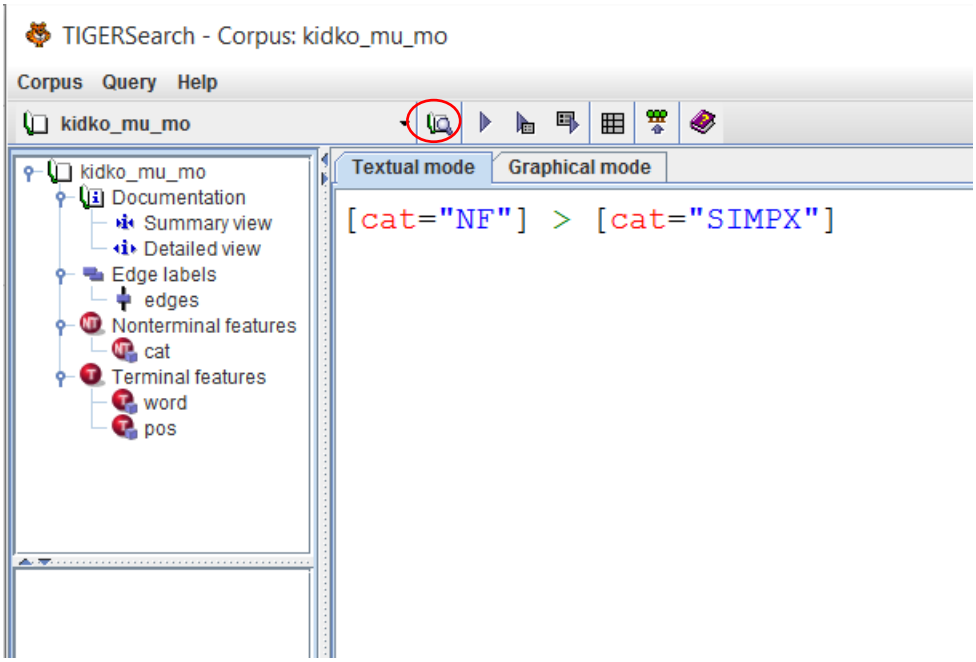
1.1.3 Arbeiten mit TIGERSearch

Um eine Suche zu starten, kann entweder das Textfeld oder die graphische Oberfläche genutzt werden. Im Textfeld wird die TIGERSearch Query Language (s.u.) genutzt.

Beispiel 1: *Suche nach einem Nebensatz im Nachfeld.*

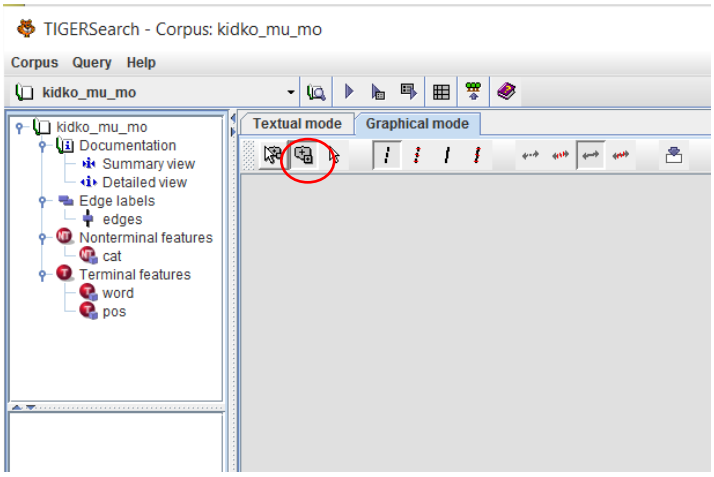
Textoberfläche

1. Eingabe der Suchanfrage in das Textfeld in der TIGERSearch query language
2. Klick auf den Suchbutton

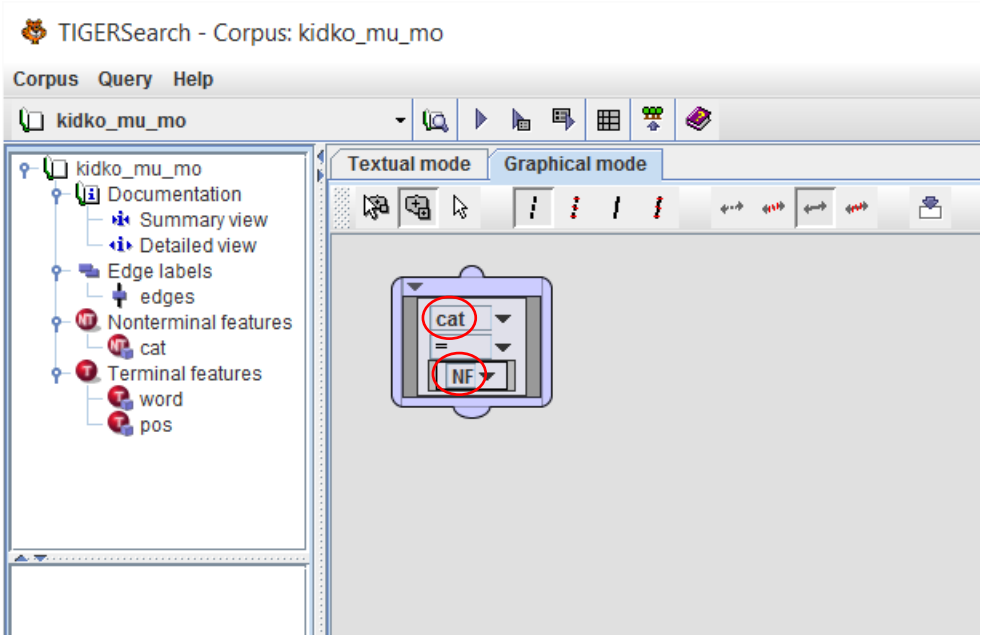


Graphische Oberfläche:

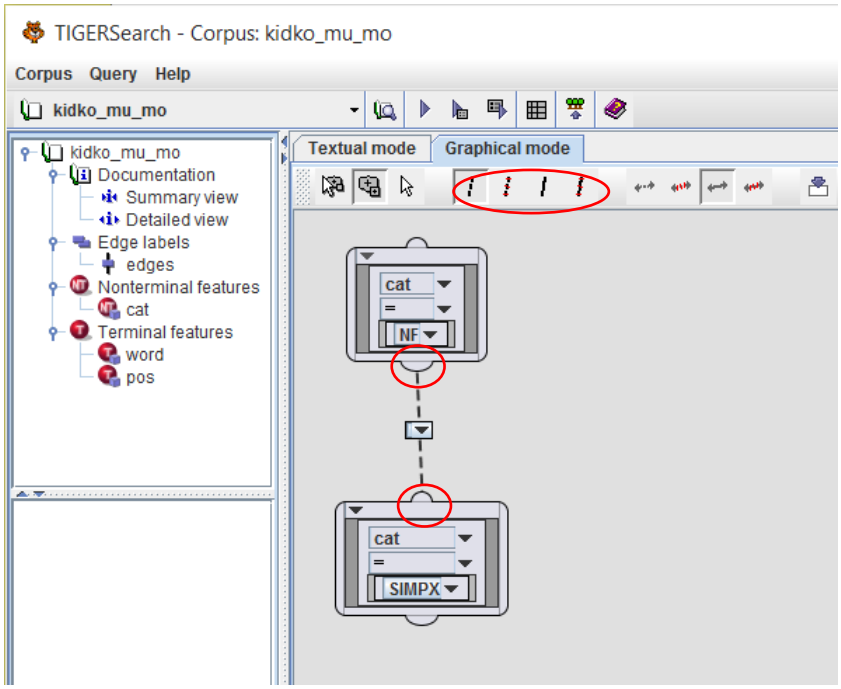
1. neuen Knoten aktivieren



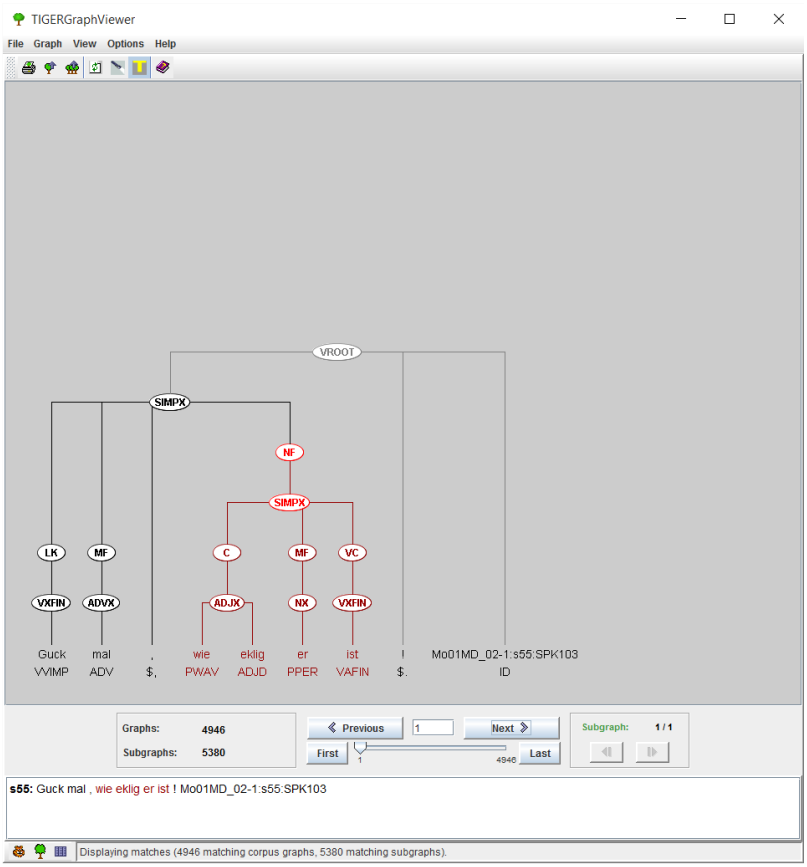
2. neuen Knoten durch Doppelklick einfügen und CAT/POS mit entsprechenden Tags auswählen (durch Klick auf die Schaltfläche und Auswahl über die Tastatur; Bsp.: NF erscheint bei Tastendruck von „n“, mehrmaliges Drücken zeigt weitere Tags)



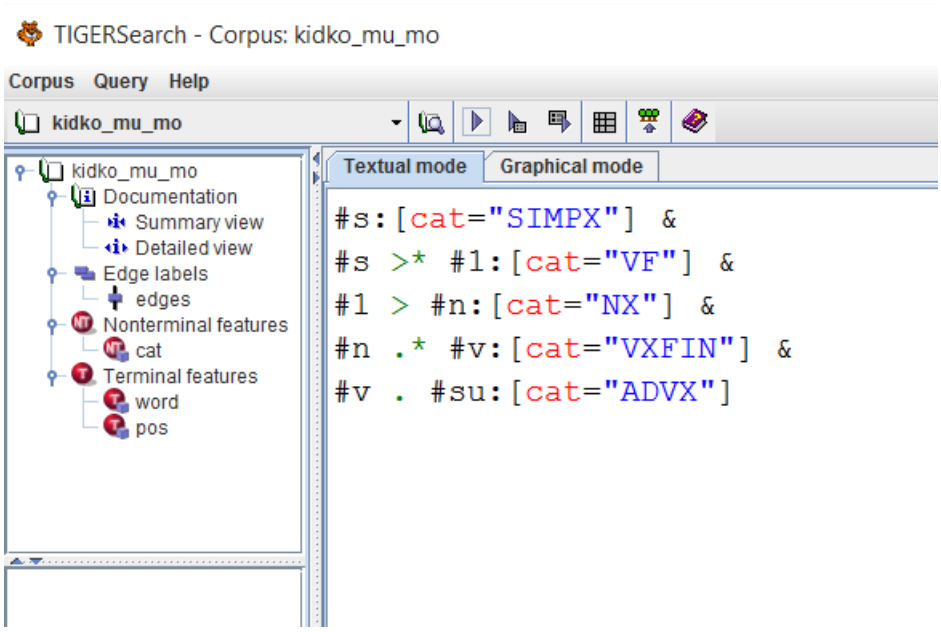
3. Einfügen des zweiten Knotens
4. Verbinden der Knoten durch Klicken auf die Schaltflächen (Dominanzverhältnis kann über entsprechende Schaltflächen verändert werden)



5. Klick auf Suchbutton



Beispiel 2: Komplexe Suche im Textfeld mithilfe von Variablen: Suche nach der Sequenz „Subjekt-finites Verb-Adverbialbestimmung“ (Zur Erklärung vgl. TIGERSearch query language)



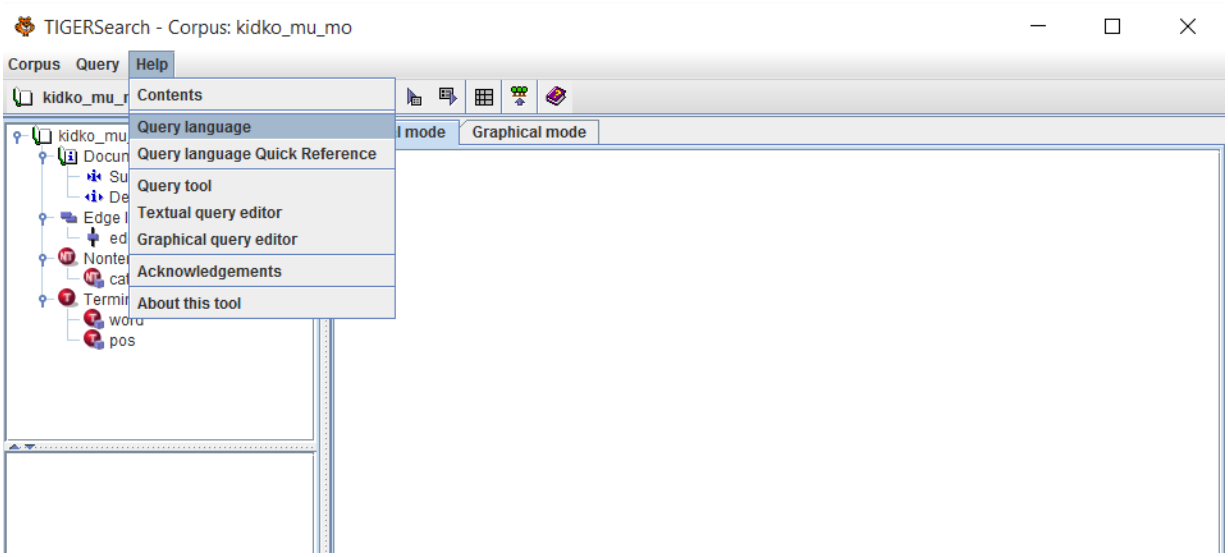
1.1.3.1 Umlaute in TIGERSearch

Aufgrund eines technischen Fehlers können die Umlaute in der aktuellen Version auf dem KidKo-PC nicht richtig dargestellt werden. Für die Suche verwenden Sie bitte folgende Zeichen:

ÃŸ = ß
Ã¤ = ä
Ã¼ = ü
Ã¶ = ö

1.1.3.2 TIGERSearch query language

Umfangreiche Dokumentationen zur Funktionsweise von TIGERSearch und der TIGERSearch Query Language sind in TIGERSearch unter „Help“ zu finden:



Die grundlegenden Funktionen werden im Folgenden erläutert.

TIGERSearch ermöglicht die Suche nach Wörtern (*word*), Wortarten (*pos*), minimalen Phrasen (*cat*) und topologischen Feldern (*cat*). Gesucht wird immer innerhalb von Sätzen und nicht satzübergreifend. Alle Merkmale, die in der Suchanfrage verbunden werden, müssen also innerhalb eines Satzes zutreffen Eine Suchanfrage besteht aus mindestens einem dieser Elemente mit dem entsprechenden *Wert*, beginnt stets mit einer geöffneten, eckigen Klammer und endet mit einer geschlossenen eckigen Klammer. *Werte* sind die verschiedenen Kategorien innerhalb der Klassen pos und cat. Pos beinhaltet POS-Tags auf Grundlage des STTS, beispielsweise also ADJA für attributives Adjektiv, NE für Eigennamen oder PPER für Personalpronomina. Alle pos-Tags können in TIGERSearch unter der Schaltfläche „pos“ in der linken Menüauswahl eingesehen werden. Cat beinhaltet die topologischen Felder wie „VF“ („Vorfeld“), „MF“ (Mittelfeld) oder „VC“ (Verbkomplex/Rechte Satzklammer). Eine Übersicht kann ebenfalls im linken Menü aufgerufen werden. Sowohl pos-, als auch cat-Tags weisen im KiDKo Besonderheiten auf, beispielsweise wurde das Feld „LA“ (Linkes Außenfeld) eingeführt, um Adverbialbestimmungen in Adverbialbestimmung-Subjekt-finites Verb-Strukturen zu taggen. Eine Übersicht der im KiDKo verwendeten POS-Tags befindet sich unter dem Punkt Tags im Kiezdeutschkorpus.

Eine minimale Suchanfrage könnte z.B. folgendermaßen aussehen:

[pos=“PPER“] sucht alle Personalpronomina im gesamten Korpus

Es ist möglich „entweder-oder“-Suchanfragen zu stellen, in dem das Zeichen „|“ benutzt wird:

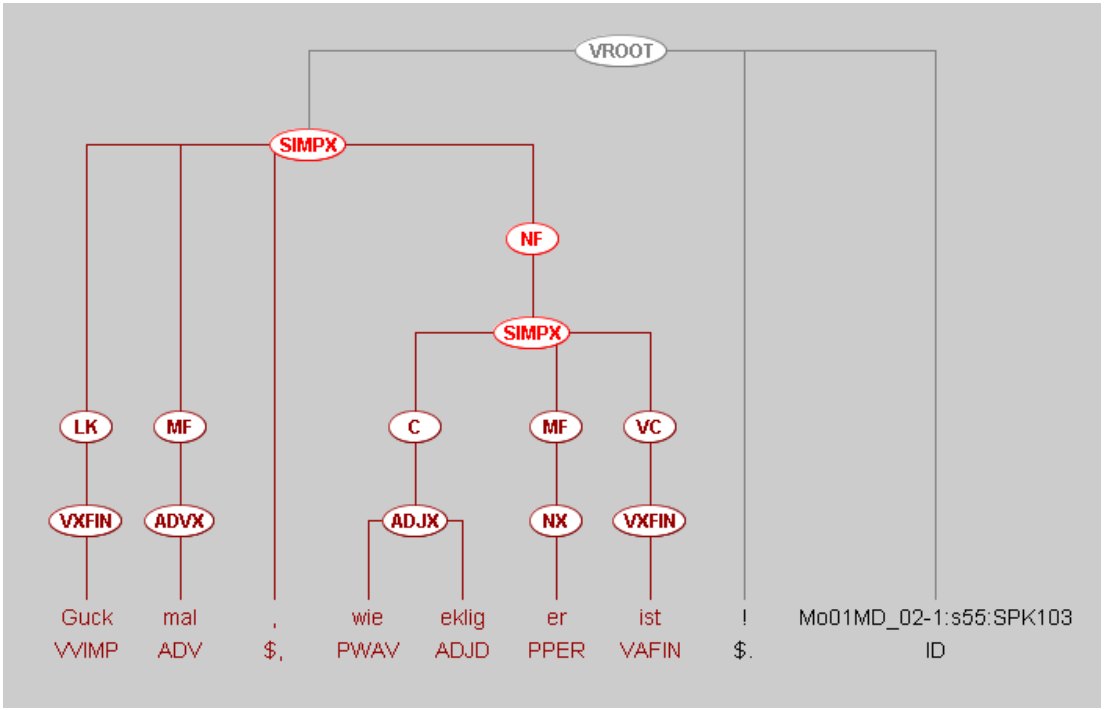
[pos = ("ADJA" | "ADJD")] Sucht alle Vorkommen von ADJA oder ADJD

Ein Element kann durch mehrere Werte spezifiziert werden:

[word="Kind" & pos="NN"]

Die TIGERSearch query language erlaubt Suchanfragen, bei der die Dominanz von Knoten spezifiziert werden kann(Knoten sind beispielsweise bspw. topologische Felder bei komplexen Sätzen). Dadurch ist es beispielsweise möglich, nach Nebensätze im Nachfeld eines Satzes zu suchen. In der folgenden Suchanfrage soll ein Satzknoten einen Nachfeldknoten dominieren. Dieser Nachfeldknoten wiederum soll einen neuen Satzknoten, der den Nebensatz beinhaltet, dominieren.

```
[cat="SIMPX"] > #nf:[cat="NF"] &
#nf > [cat="SIMPX"]
```



Zudem können Sequenzen von Wörtern/Wortarten oder Feldern gesucht werden. Die wichtigsten Operationen sind „>“ für Dominanz und „.“ für Folge. Beide Operatoren können mithilfe von „*“ modifiziert werden. „>*“ bezeichnet eine Dominanz mit einer beliebigen Anzahl an Knoten zwischen zwei Elementen. „.*“ bezeichnet die Folge zweier Einheiten mit einer beliebigen Anzahl an Elementen dazwischen.

Beispiele:

[cat="VF"] > [cat="NX]	sucht ein Vorfeld („VF"), das ein nominales Element, wie ein Nomen („NN") oder einen Eigennamen („NE"), („NX") direkt dominiert
[cat="SIMPX"] >* [cat="NX"]	sucht einen Satzknoten, der ein NX beinhaltet
[word="am"] . [word="liebsten"]	sucht Wortfolgen von „am liebsten" (= „am", unmittelbar gefolgt von „liebsten")
[cat="VF"] .* [cat="NF"]	sucht ein VF, auf das in der Struktur ein Nachfeld folgt

Einige Zeichen sind der Suchanfragesprache vorenthalten und müssen mit „\" als Suchanfrage gekennzeichnet werden:

```
[word="\."]
```

Um eine *komplexe Suchanfrage* wie in Beispiel 2 zu stellen, müssen zwei Punkte beachtet werden:

1. Es kann jeweils nur eine Operation („>", „.") zwischen zwei Elementen ausgedrückt werden.
2. Jedes Element muss durch eine *Variable* kodiert werden.

Variablen werden durch „#" + einem beliebigen Zeichen + „:" definiert.

```
#satz:[cat="SIMPX"]
```

Variablen können durch Operatoren wie gewohnt verbunden werden. Für eine komplexe Suchanfrage legt man zuerst die einzelnen Variablen fest und setzt sie anschließend zueinander ins Verhältnis:

Beispiel: Deklarativsatz mit der Sequenz: „Subjekt - finites Verb – Adverbialbestimmung"

```
#s:[cat="SIMPX"] &
#vf:[cat="VF"] &
#nx:[cat="NX"] &
#v:[cat="VXFIN"] &
#a:[cat="ADVX"] &
#s >* #vf &
#vf > #nx &
#nx . #v &
#v . #a &
#s > [word="\."]
```

Die Anfrage muss nicht in dieser Reihenfolge gestellt werden. Variablen können auch während einer Operation definiert werden:

```
#s:[cat="SIMPX"] &
```

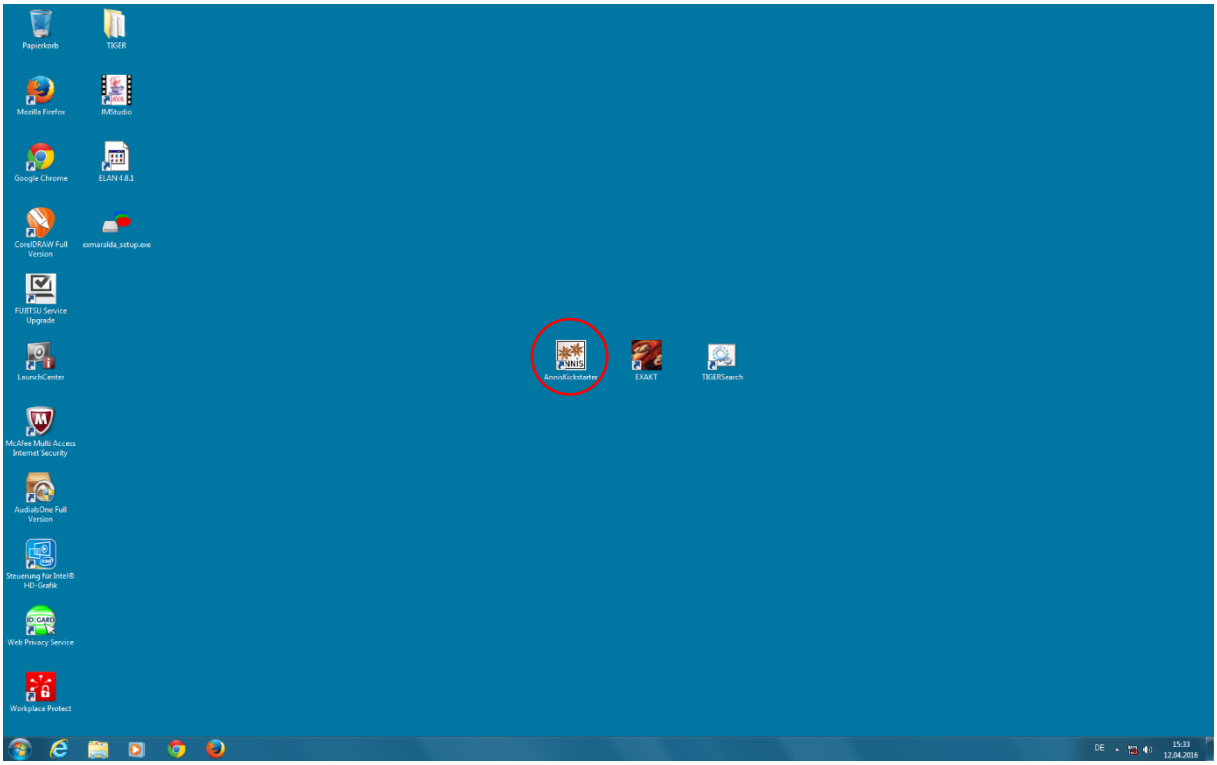
```
#s >* #vf:[cat="VF"] &
#vf > #nx:[cat="NX"] &
#nx . #v:[cat="VXFIN"] &
#v . #a:[cat="ADVX"] &
#s > [word="\."]
```

In der Suche können auch reguläre Ausdrücke benutzt werden (vgl. ANNIS & reguläre Ausdrücke: Einführung).

1.2 ANNIS

1.2.1 ANNIS starten

Das Programm wird durch Klicken des Icons auf dem Desktop gestartet.

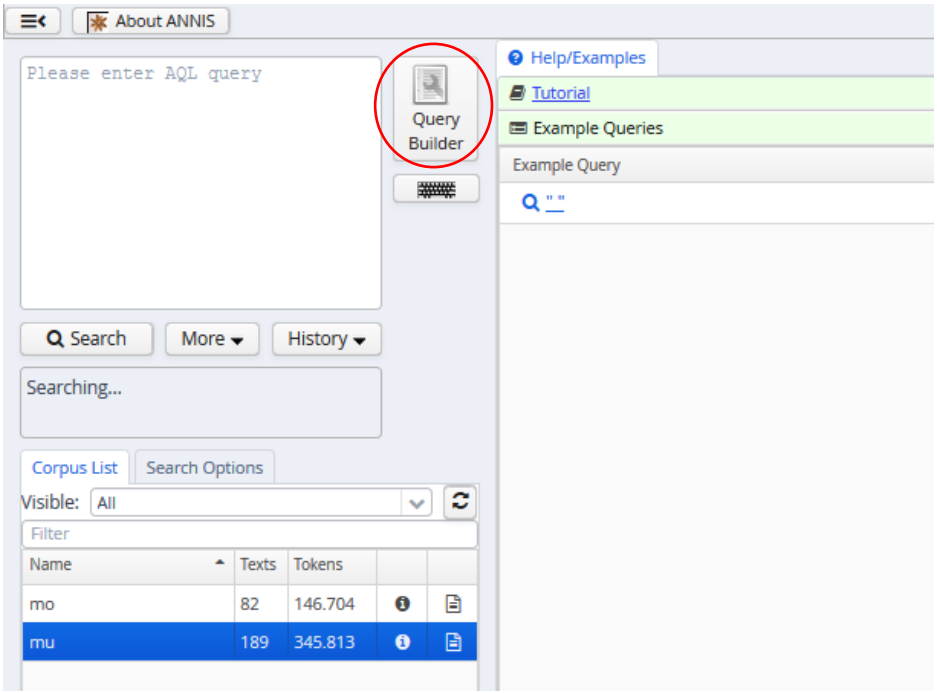


1.2.2 Arbeiten mit ANNIS

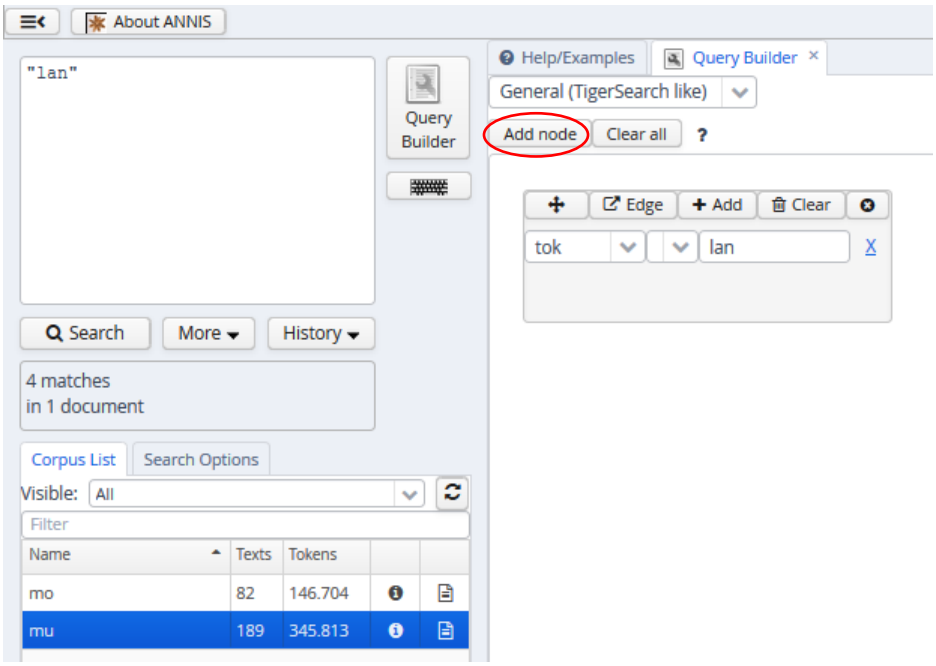
Im folgenden Abschnitt werden die wichtigsten Informationen dargestellt, die für eine Suche in ANNIS notwendig sind. Wie in TIGERSearch kann für die Suche in ANNIS entweder die Textoberfläche mithilfe der ANNIS Query Language – AQL oder der Query Bilder benutzt werden.

1.2.2.1 Query Bilder

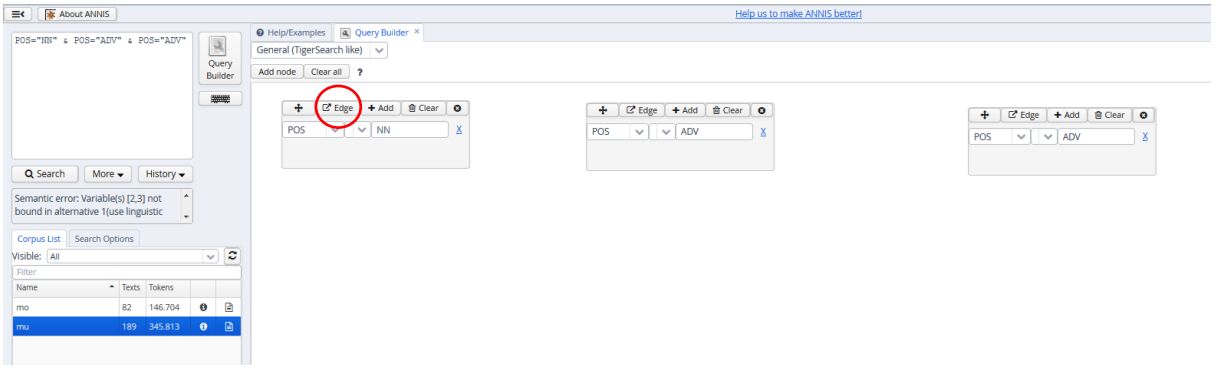
1. Der Query Builder wird durch Klicken auf das entsprechende Symbol geöffnet.

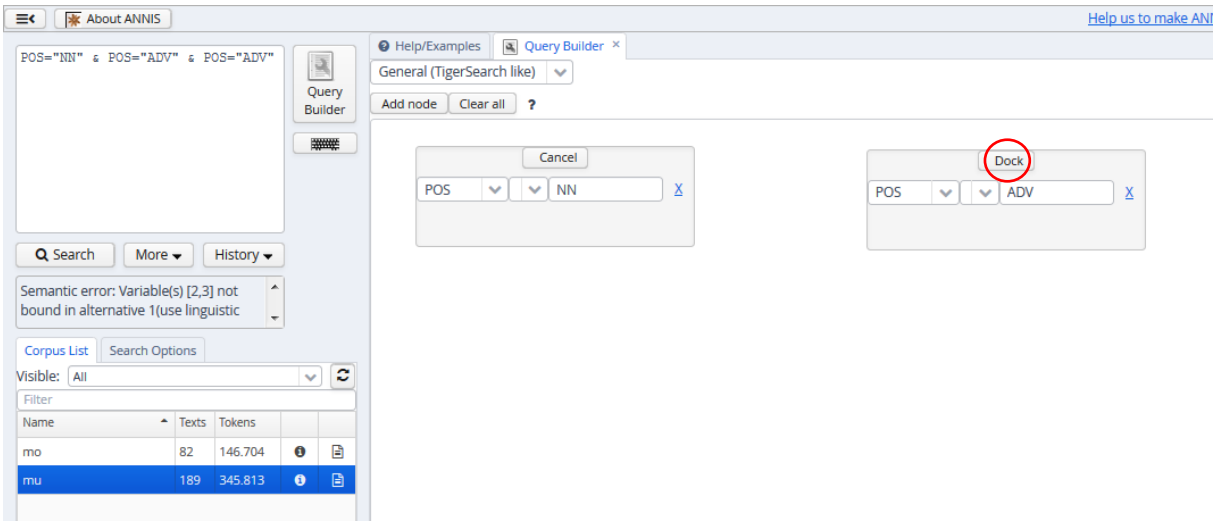


2. Alle Knoten (d.h. Elemente wie Lexeme, Wortarten oder topologische Felder), die man verwenden will, werden durch „add node“ hinzugefügt. Ein Knoten besteht aus einem Attribut-Wert-Paar. Attribute (beispielsweise „tok“ für Token, „pos“ für Wortart) und Werte (beispielsweise Lexeme oder pos-Tags) können über die Menüs im Knotenfeld eingegeben werden. Das folgende Bild zeigt beispielsweise die Suche nach dem Lexem „lan“. Das Attribut ist hier „tok“ (für Token), der Wert „lan“. Alle Nomen ließen sich beispielsweise mit dem Attribut-Wert-Paar pos=“NN“ finden („pos“ für Wortart, „NN“ für Nomen). Eine Übersicht über Attribute und Werte kann der Übersicht Tags im Kiezdeutschkorpus entnommen werden.



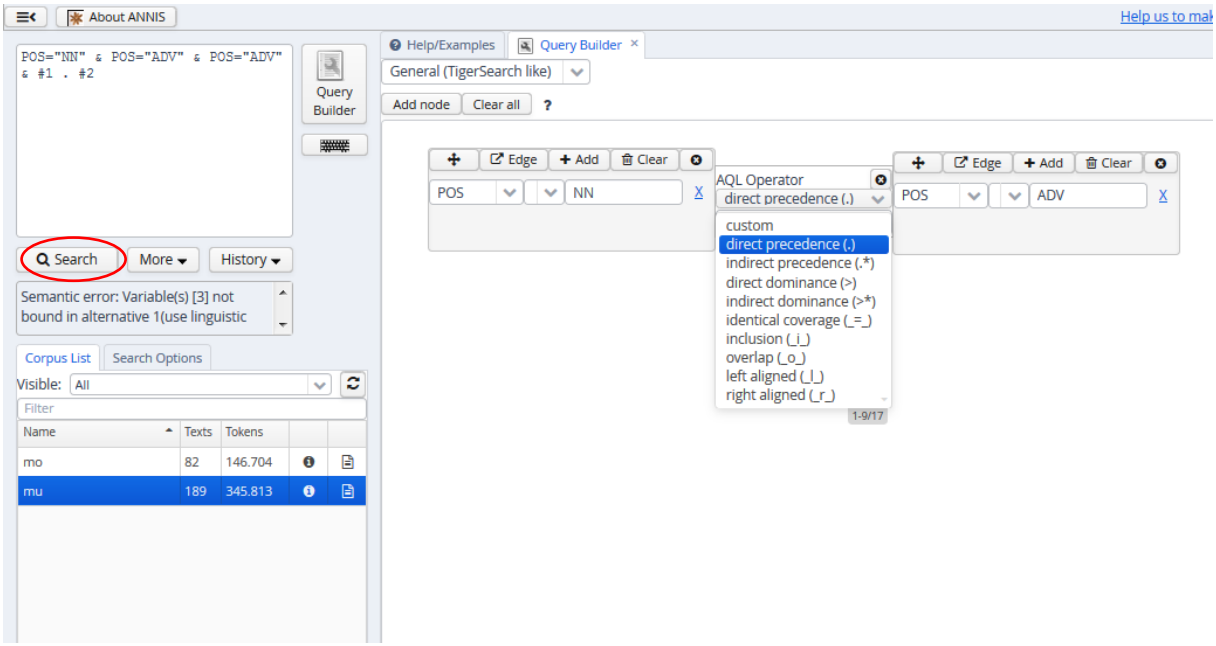
3. Durch Klicken auf „Edge“ kann die Relation zwischen zwei Elementen angegeben werden. Per Klick auf „Dock“ wird die Relation etabliert.





4. Die Relation kann per Menüauswahl spezifiziert werden. Für KiDKo sind dabei folgende Operatoren relevant:

.	direkte Folge zweier Elemente
.*	indirekte Folge (eine beliebige Menge an Elementen zwischen zwei Elementen)
>	direkte Dominanz eines Elements auf ein anderes
>*	indirekte Dominanz (eine beliebige Menge an Elementen zwischen zwei Elementen)



5. Klick auf „Search“ liefert das Suchergebnis.

1.2.2.2 ANNIS Query Language – AQL

ANNIS basiert auf einem Modell, das, wie TIGERSearch, Knoten und Verbindungen zwischen den Knoten annimmt. Eine Suchanfrage besteht entweder aus einem Token (“Hund“) oder einem Attribut-Wert-Paar (lemma=“Hund“). Unterschiedliche Korpora können unterschiedliche Attribut- und Wertbezeichnungen haben, da dies nicht von ANNIS, sondern von der Annotation des Korpus abhängt. Elemente müssen durch Relationen („>“, „.“, „>*“, „.*“) in Verbindung gesetzt werden. Das Attribut muss in Großbuchstaben geschrieben werden, Klammern sind nicht nötig

POS = „NN“	findet alle Nomen
POS=“NN“ .* POS”VVFIN”	Findet alle Vorkommen eines Nomens, das in der Äußerung von einem finiten Verb gefolgt wird.

Wie bei TIGERSearch werden komplexe Relationen durch die Setzung von Variablen gewährleistet. Dazu werden zunächst alle Elemente aufgelistet, diesen werden dann Variablen zugewiesen, die zu einander in Relation gesetzt werden:

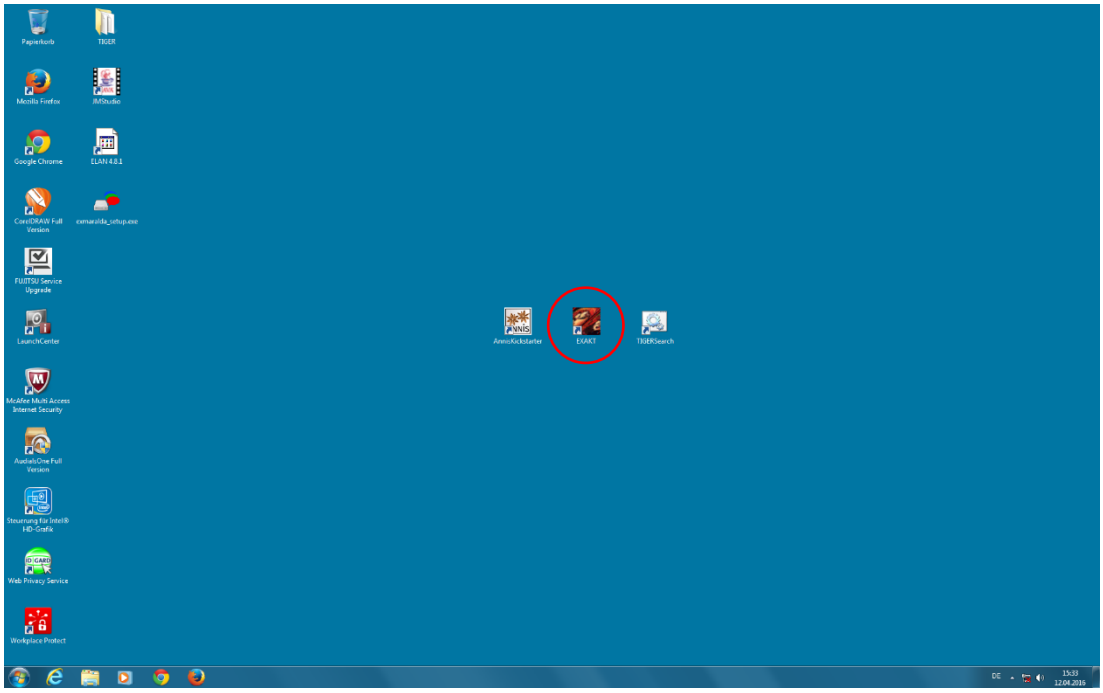
POS=”NN” & POS=“ADV“ & POS =“ADV“ & #1 . #2 & #2 .* #3	Findet ein Nomen, dem ein Adverb direkt folgt. Diesem folgt in der Äußerung ein weiteres Adverb.

Weitere Informationen zu ANNIS und Suchbeispiele befinden sich unter: ANNIS & reguläre Ausdrücke: Einführung.

1.3 EXAKT (EXMARaLDA)

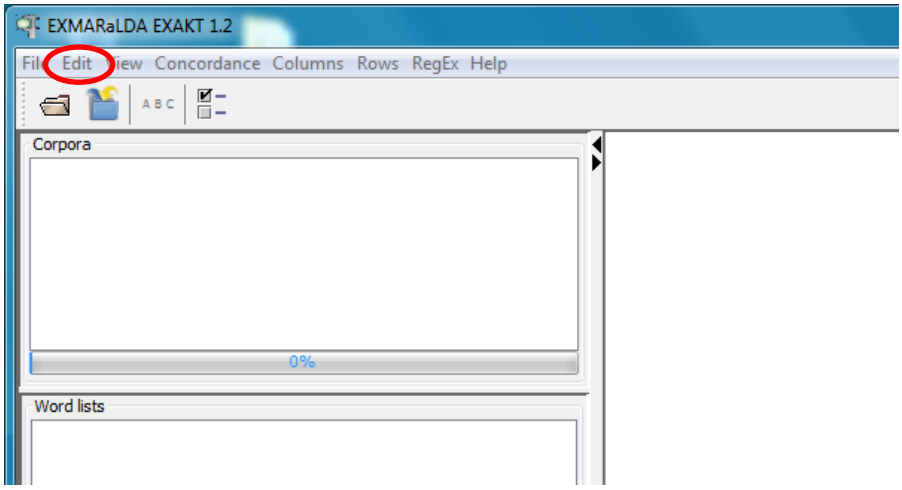
1.3.1 Exakt starten

EXAKT wird per Klick auf das Symbol auf dem Desktop gestartet.

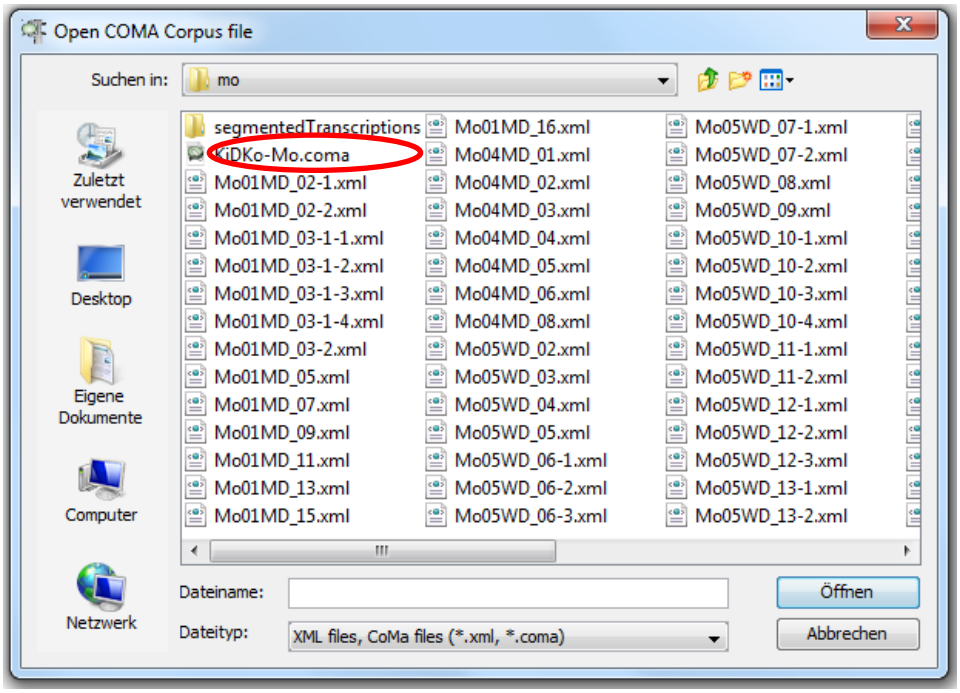


1.3.2 Korpus öffnen

1. EXAKT öffnen
2. File > Open Corpus



3. Im Auswahlmenü („Suche in“) „exmaralda“ auswählen
4. mu- oder mo-Korpus auswählen
5. „*.coma“-Datei auswählen



1.4 Arbeiten mit EXAKT

Eine Dokumentation zur Nutzung von EXAKT bietet das Quickstart-Tutorial der EXMARaLDA-Entwickler Thomas Schmidt und Kai Wörner:

http://www.exmaralda.org/pdf/Quickstart/EXAKT/Quickstart_working_with_EXAKT_DE.pdf

1.5 Tags im Kiezdeutschkorpus

POS-Tags (Attribut: pos)
Alle blau markierten Tags sind neu für das KiDKo entwickelt worden. Eine genauere Beschreibung der übrigen Tags bietet das Institut für Maschinelle Sprachverarbeitung der Universität Stuttgart:

http://www.ims.uni-stuttgart.de/forschung/ressourcen/lexika/TagSets/stts-table.html

Beschreibung	**Tag/Wert**	
Adjektive	ADJ	
- attribuierend	ADJA	
- adverbial/prädikativ	ADJD	
Adverbien	ADV	
Adpositionen		
- Präposition, Zirkumposition links	APPR	
- Präposition mit Artikel (Verschmelzung)	APPRART	
- Postposition	APPO	
- Zirkumposition rechts	APZR	
- Bestimmter oder unbestimmter Artikel	ART	
Kardinalzahl	CARD	
Fremdsprachliches Material	FM	
Konjunktionen		
- Vergleichskonjunktion	KOKOM	
- Nebenordnende Konjunktion	KON	
- Unterordnende Konjunktion mit „zu“ und Infinitiv	KOUI	
- Unterordnende Konjunktion mit Satz	KOUS	
Eigennamen	NE	
Normales Nomen	NN	
Pronomen		
- Substituierendes Demonstrativpronomen	PDS	
- Attribuierendes Demonstrativpronomen	PDAT	
- Substituierendes Indefinitpronomen	PIS	
- Attribuierendes Indefinitpronomen	PIAT	
- Irreflexives Personalpronomen	PPER	

- Substituierendes Possessivpronomen	PPOSS	
- Attribuierendes Possessivpronomen	PPOSAT	
- Substituierendes Relativpronomen	PRELS	
- Attribuierendes Relativpronomen	PRELAT	
- Reflexives Personalpronomen	PRF	
- Substituierendes Interrogativpronomen	PWS	
- Attribuierendes Interrogativpronomen	PWAT	
- Adverbiales Interrogativ- oder Relativpronomen	PWAV	
Pronominaladverb	PROAV	
Partikeln		
- ``zu" vor Infinitiv	PTKZU	
- abgetrennter Verbzusatz	PTKVZ	
- Negationspartikel	PTKNEG	
- Antwortpartikel	PTKANT	
- Rezeptionspartikel	PTKRZ	
- Fillerpartikeln, gefüllte Pausen	SPFILL	**äh**
- Gliederungspartikeln	SPINI	**Na** mal schauen
- Interjektion	SPITJ	**Ey** !
- Onomatopoetika	SPONO	und das machte **schepper**
- Fragepartikel	SPQU	**Bitte** ?
- Antwortpartikel	SPRS	**Bitte** !
Verben		
- Finites Vollverb	VVFIN	
- Imperativ, Vollverb	VVIMP	
- Infinitiv, Vollverb	VVINF	
- Infinitiv mit „zu“, Vollverb	VVIZU	
- Partizip perfekt, Vollverb	VVPP	
- finites Hilfsverb	VAFIN	
- Imperativ, Hilfsverb	VAIMP	
- Infinitiv, Hilfsverb	VAINF	
- Partizip Perfekt, Hilfsverb	VAPP	
- finites Verb, Modalverb	VMFIN	
- Infinitiv, Modalverb	VMINF	
- Partizip Perfekt, Modalverb	VMPP	
- Inflektive	VVINFL	Kaffeetasse heb

Kompositionsglied	TRUNC	
Platzhalterpartikel	DINGS	`Das Buch auf dem **Dings**`
Nichtwörter	XY	
- Abbrüche	XYB	`Tuncay ist ja **ha** (-) ist ja auch weg jetzt`
- Unverständliches Material	XYU	`Er hat **UNINTERPRETABLE** gefragt .`
Interpunktion		
- Punkt, Ausrufezeichen, Fragezeichen	$.	
- Komma	$,	
- Klammer	$(	
- Abbruch	$#	

CAT-Tags (Attribut: „cat")

Chunks (Oberkategorien)

Beschreibung	**Tag/Wert**	
Adjektive	ADJX	
Adverbien	ADVX	
Gesprächspartikeln	DM	
Fragmente	FRAG	
Fremdsprachliche Material	FX	
Nominale Elemente	NX	
Nominale Elemente mit gestrandetem Determinierer	sNX	`(**sNX der**) (NX seinen Sohn) (NX liebende Vater)`
Präpositionale Elemente	PX	
Satz	SIMPX	
Finite Verben	VXFIN	
Infinite Verben	VXINF	

Topologische Felder

Beschreibung	**Tag/Wert**
Koordinationsfeld	KOORD
Koordinationsfeld mit V2	PAROD
Freies Thema	FT
Linksversetzung	LV
Linkes Außenfeld	LA
Vorfeld	VF
Komplementierer mit Verbletzt	C

Linke Klammer	LK
Mittelfeld	MF
Verbalkomplex	VC
Ersatzinfinitiv	VCE
Nachfeld	NF
Rechtes Außenfeld	RA
Felderkoordination	FKOORD/FKONJ
Parenthesen	PAR

2 ANNIS & reguläre Ausdrücke: Einführung

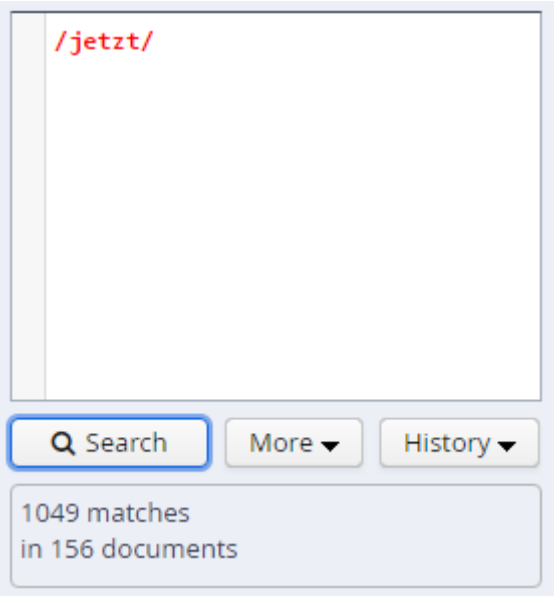
2.1 Annotationsebene in KiDKo

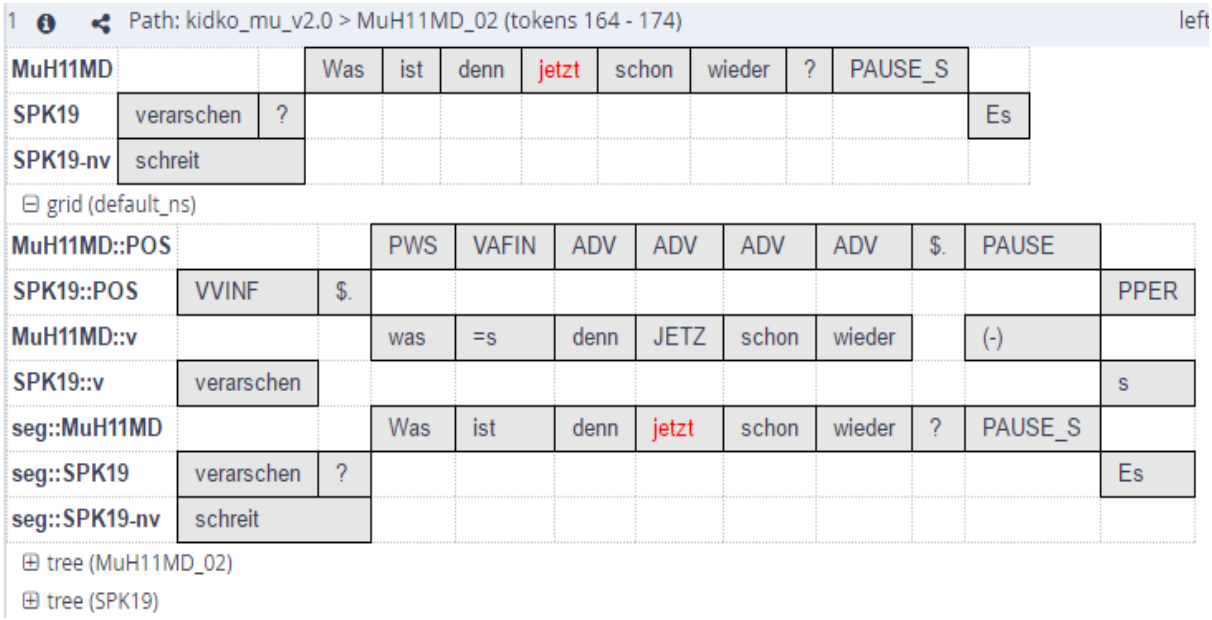
Die Daten im KiDKo sind auf mehreren Ebenen annotiert.

Eingabe	Beschreibung	Beispiele
n	normalisierte Ebene	nichts, Ja, nachgezahlt
v	Transkriptionsebene (verbale Ebene, nicht normalisiert)	jetz, nix, RICHtig, =m
nv	nonverbale Ebene	lacht, schreit, genervt
POS	Wortartebene (part-of-speech)	VVINF, ADJA
tr	türkische Transkriptionsebene (nicht normalisiert)	kapışak, okicaksan
trnorm	türkische Normalisierung	mı, okuyacaksan
trdtue	Deutsche Übersetzung (frei)	Der Mann kommt nicht mehr.
trdtwwue	deutsche Übersetzung (Wort für Wort)	die Frau, kommt, Interjektion, aus dem Kopf

Im Korpus lässt sich auch nach bestimmten Sprechern oder Transkripten suchen, z.B. SPK101, MuH11MD, usw.

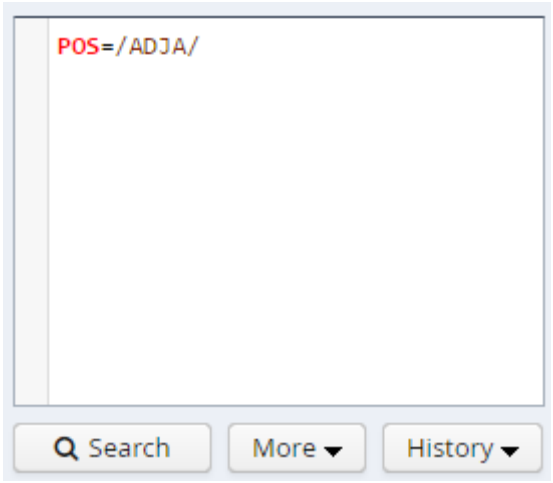
Wenn Sie ein Wort auf der normalisierten Ebene suchen wollen, geben Sie das Wort in Schrägstrichen ein. Wenn Sie keine Ebene spezifizieren, sucht ANNIS automatisch auf der normalisierten Ebene.

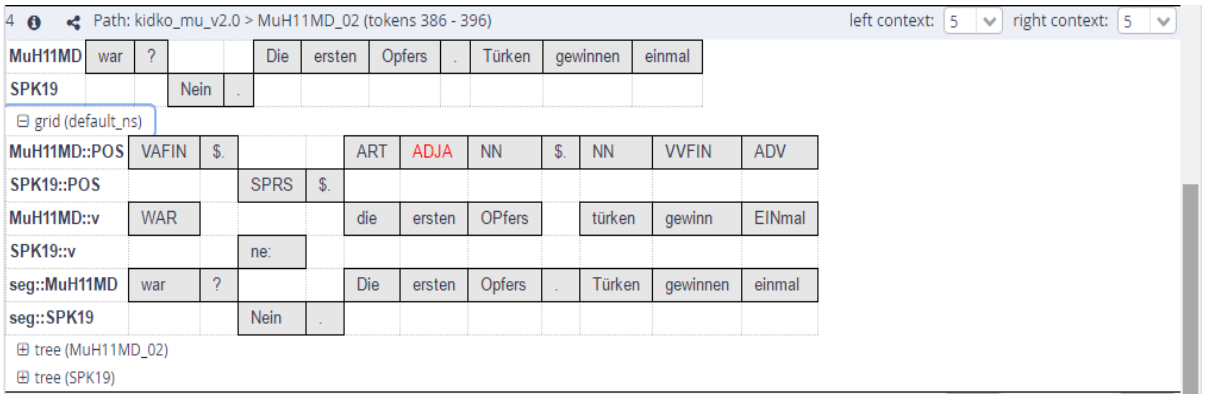


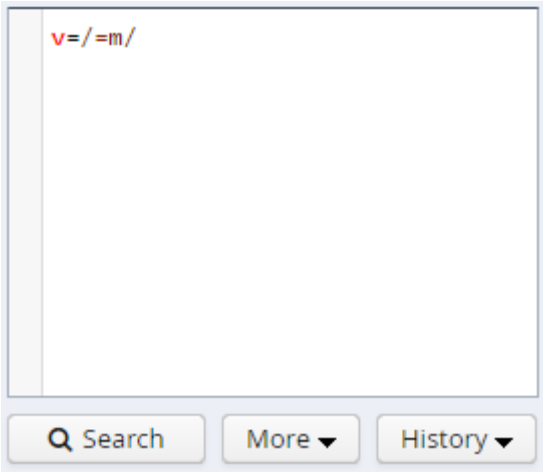


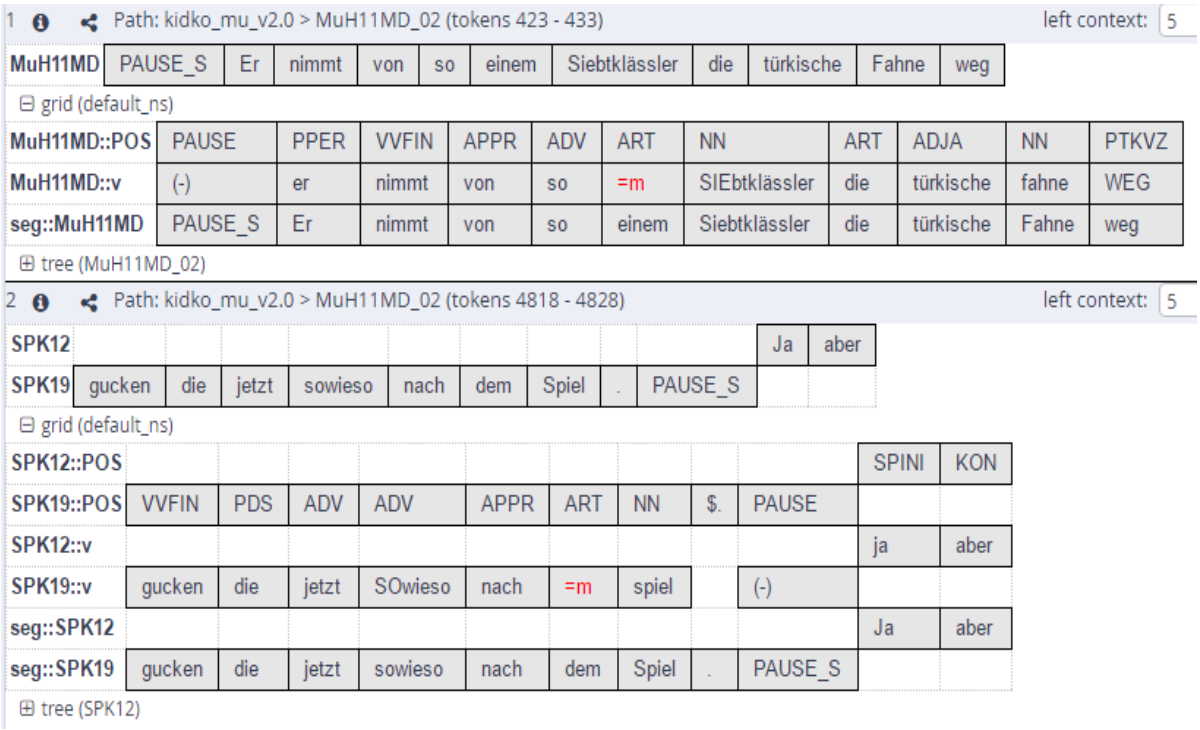
Wenn Sie eine andere als die normalisierte deutsche Ebene durchsuchen wollen, wie z.B. **POS** oder **v**, müssen Sie dies spezifizieren:

`POS=/ADJA/` oder `v=/=m/`.

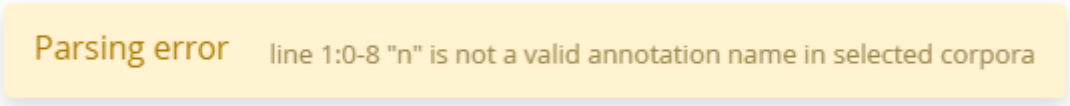








Achtung: Die normalisierte Ebene **n** ist vom Programm als Default-Ebene definiert. Deswegen erzeugt die Eingabe einer Spezifizierung auf dieser Ebene einen Fehler. Bei der Eingabe von `n=/jetzt/` tritt daher dieser Fehler auf.

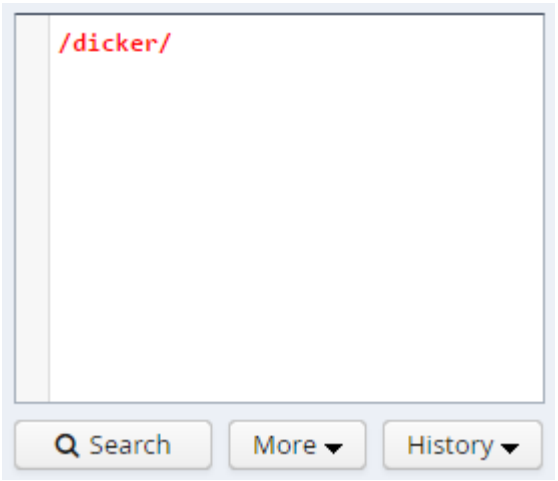


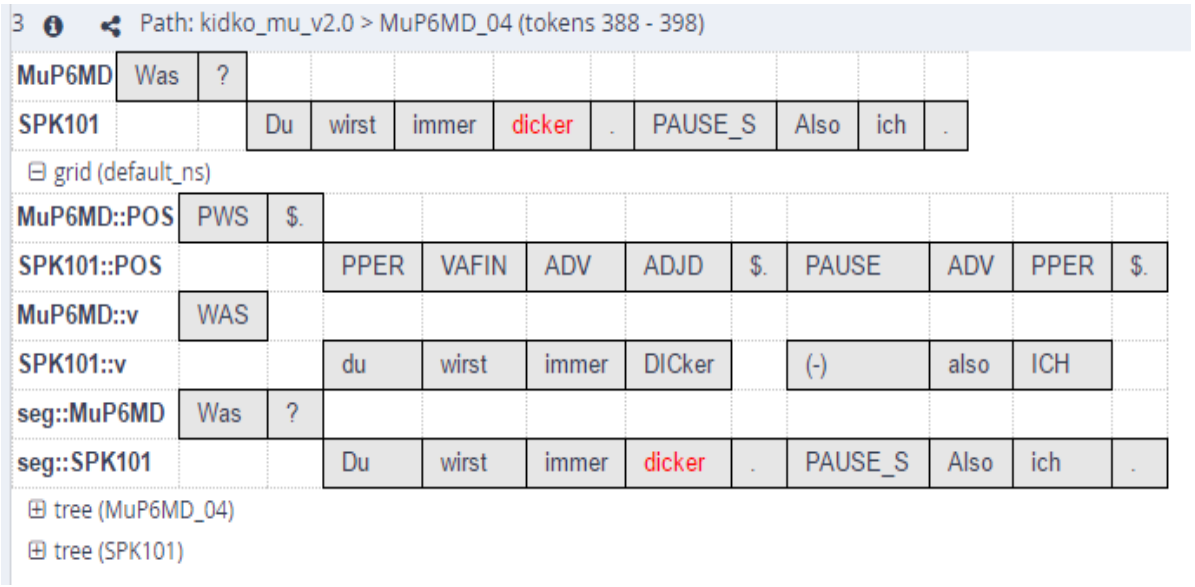
Bei allen anderen Ebenen muss eine Spezifizierung vorgenommen werden (vgl. S. 20)

2.2 ANNIS & reguläre Ausdrücke: Einführung

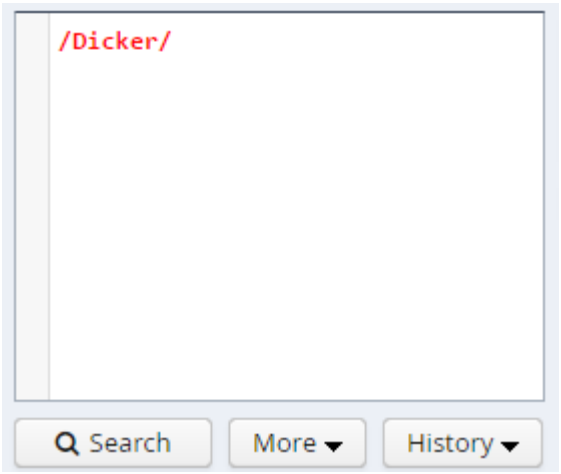
2.2.1 Groß- und Kleinschreibung

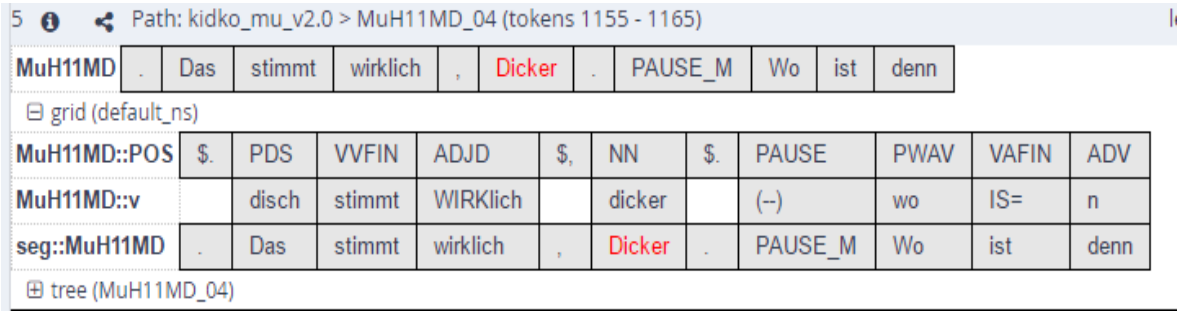
Wenn Sie `/dicker/` auf der Normalisierungsebene suchen, werden alle Adjektive als Treffer ausgegeben, da *dicker* hier als Adjektiv kleingeschrieben ist.



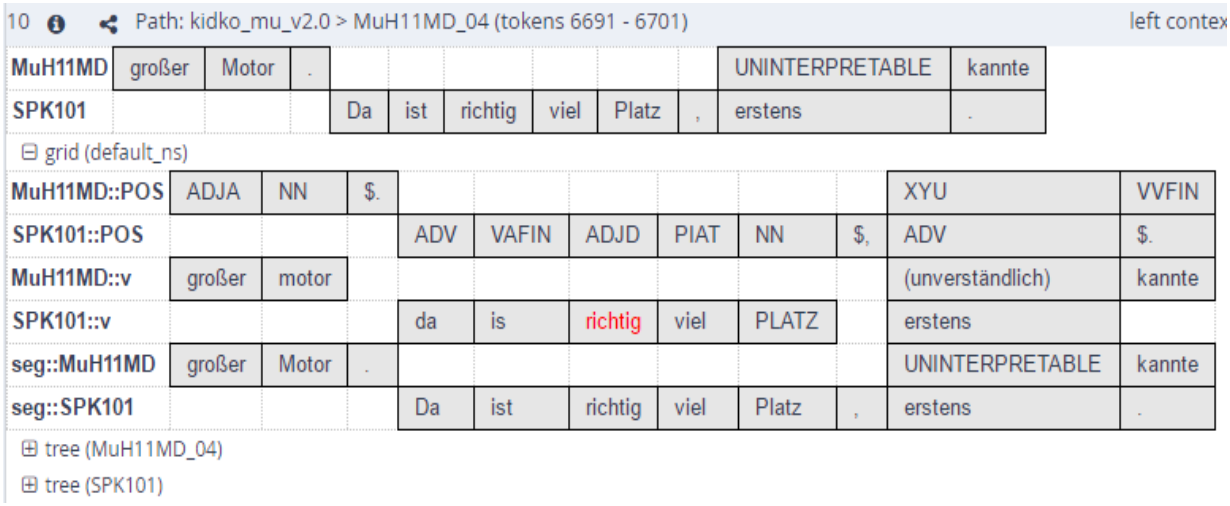


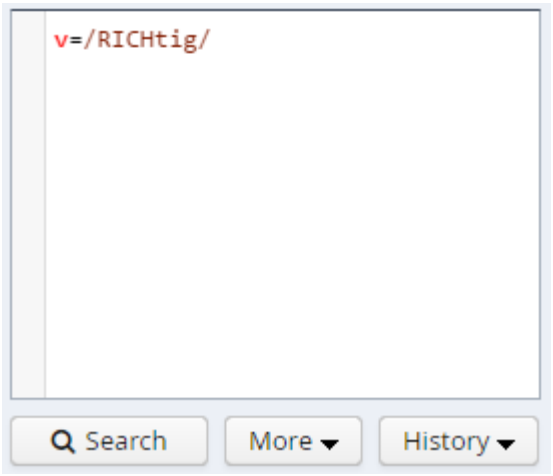
Wenn Sie `/Dicker/` auf der Normalisierungsebene suchen, werden als Treffer alle Nomen ausgegeben, da *Dicker* hier als Nomen großgeschrieben ist (außerdem werden Adjektive ausgegeben, die am Satzanfang stehen).

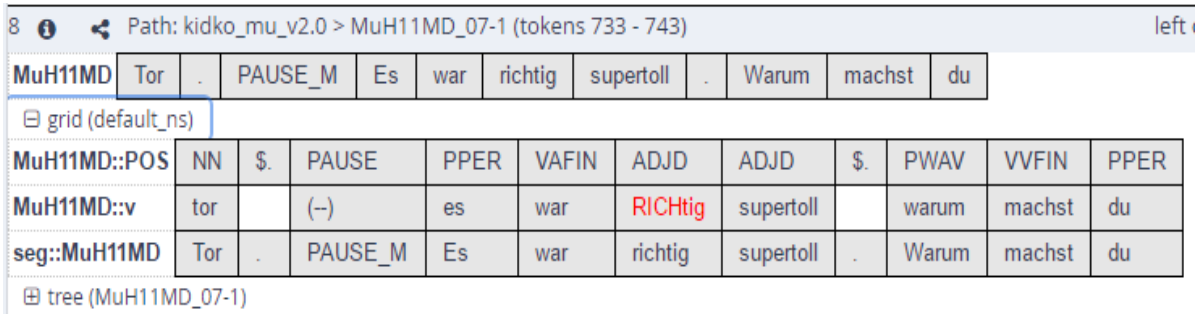




Beachten Sie: Wenn Sie die verbale Ebene v durchsuchen, markiert Großschreibung, anders als in der normalisierten Ebene, Betonung (ausführlich in den Transkriptionsrichtlinien für KiDKo). Wenn Sie `v=/richtig/` suchen, werden Sie daher andere Treffer finden als bei der Suche nach `v=/RICHtig/` (siehe unten).



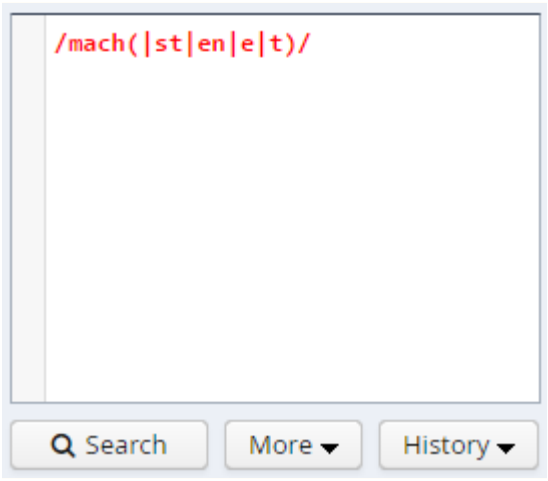




2.2.2 Entweder / oder

Wenn Sie sowohl `/zum/` als auch `/zur/` suchen möchten, können `/zu[m|r]/` suchen.

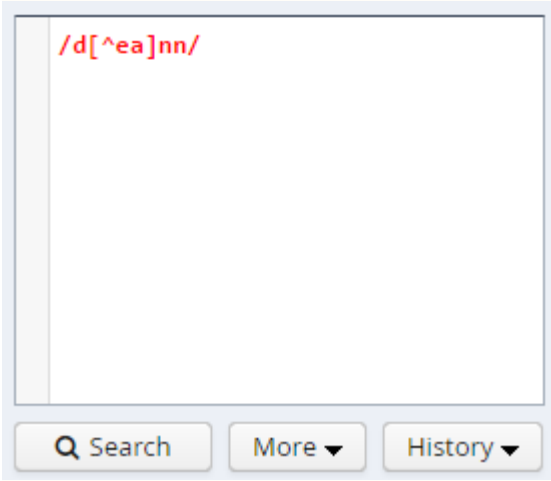
Ein weiteres Beispiel ist folgende Suchanfrage: `/mach(|st|en|e|t)/` Diese Eingabe findet *mach*, *machst*, *machen*, *mache* und *macht*.



2.2.3 Ausschließen von Zeichen

Wenn Sie ein oder mehrere Zeichnen ausschließen wollen, können Sie „^" benutzen. Wenn Sie beispielsweise alle Wörter finden wollen, die die folgende Reihenfolge von Buchstaben aufweisen: *d*, ein Zeichen, was nicht *e* oder *a* ist, *n*, *n*, könnten Sie folgendes eingeben:

```
/d[^ea]nn/
```



2.2.4 Optionalität

Sie können auch angeben, dass etwas optional ist. Damit können Sie beispielsweise alle Beispiele von *jetzt* und *jetz* in der verbalen Ebene finden, wenn Sie Folgendes eingeben: `v=/jetz(t)?/` (Bemerkung: `?` bedeutet, dass das Element in den Klammern davor optional ist.)

3 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 688 - 698) left c

SPK19	aber	PAUSE_S	die	haben	mir	jetzt	gesagt	,	dass	es	vielleicht

⊟ grid (default_ns)

SPK19::POS	KON	PAUSE	PDS	VAFIN	PPER	ADV	VVPP	$,	KOUS	PPER	ADV
SPK19::v	aber	(-)	die	ham	mir	jetzt	geSAGT		dass	es	vielLEICHT
seg::SPK19	aber	PAUSE_S	die	haben	mir	jetzt	gesagt	,	dass	es	vielleicht

⊞ tree (SPK19)

4 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 1180 - 1190) left c

MuH11MD	.	PAUSE_L	Ich	habe	Bock	jetzt	zu	vögeln	.	Mann	!

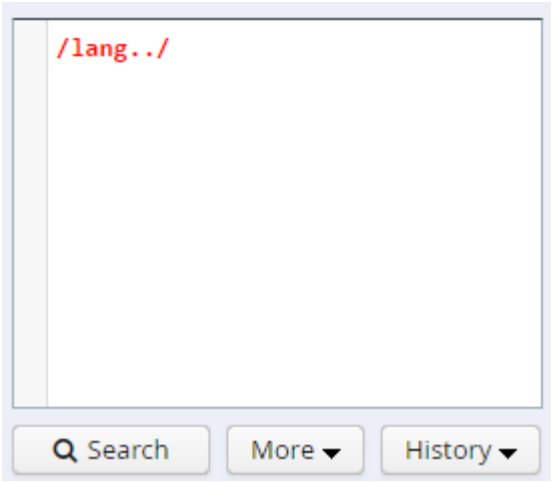
⊟ grid (default_ns)

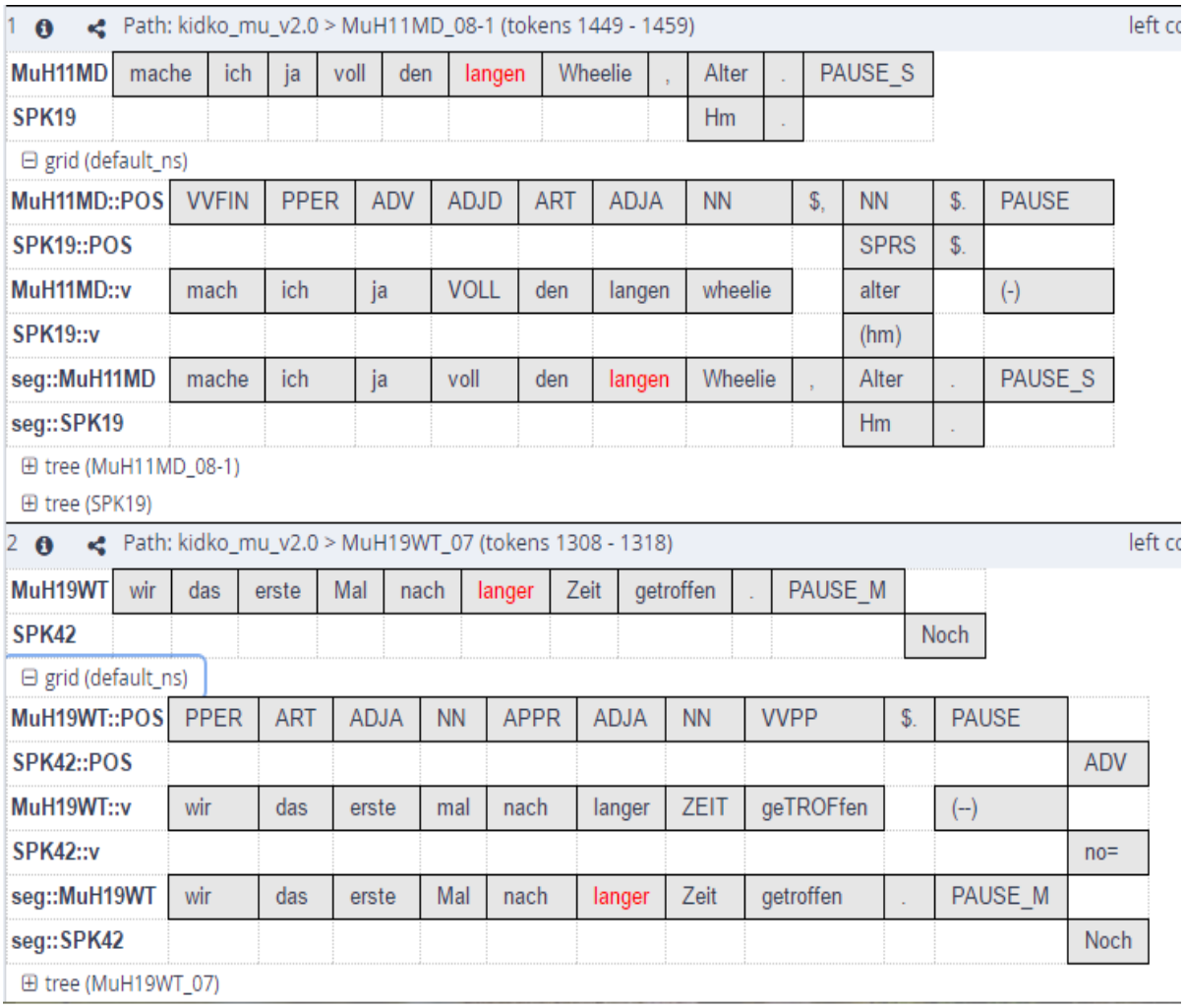
MuH11MD::POS	$.	PAUSE	PPER	VAFIN	NN	ADV	PTKZU	VVINF	$.	SPITJ	$.
MuH11MD::v		(9.2)	ich	hab	bock	jetz	zu	VÖgeln		mann	
seg::MuH11MD	.	PAUSE_L	Ich	habe	Bock	jetzt	zu	vögeln	.	Mann	!

⊞ tree (MuH11MD_02)

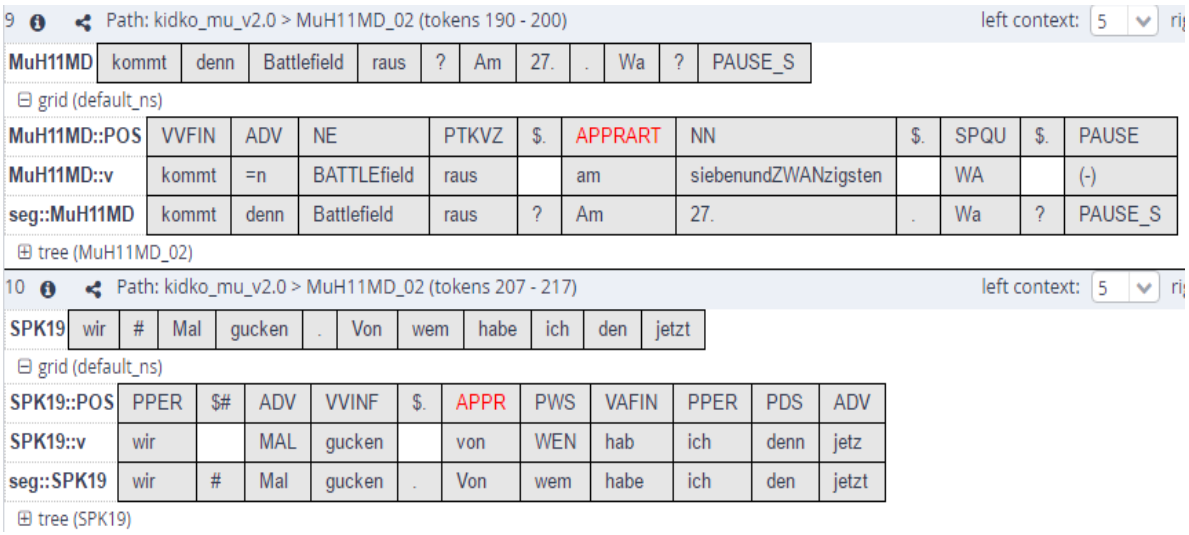
2.2.5 Ein beliebiges Zeichen

Ein Punkt kann als Platzhalter für ein beliebiges Zeichen benutzt werden. Zum Beispiel findet die Suchanfrage `/lang../` alle Wörter, die mit *lang* beginnen und noch zwei weitere Zeichen beinhalten, z.B. *langen* oder *langer*.





Sie können auch definieren, dass die Anzahl an Zeichen in einer Zeichenkette beliebig ist. So können Sie zum Beispiel alle Arten von Präpositionen (d.h. Präposition; Zirkumposition links (APPR), Zirkumposition rechts (APZR), Präposition mit Artikel (APPRART) und Postposition (APPO)) finden, wenn Sie Folgendes eingeben: `POS=/AP.*/`



Das .* kann an einer beliebigen Stelle in der Zeichenkette stehen (vorne, mittig oder hinten). Wenn Sie beispielsweise alle finiten Verben finden wollen (d.h. VVFIN (finites Verb, voll), VAFIN (finites Verb, aux) und VMFIN (finites Verb, modal)), können Sie dies folgendermaßen suchen:

```
POS=/.*FIN/
```

8 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 64 - 74)

MuH11MD	bisschen	Fett	fressen	,	dann	gehen	wir	eine	rauchen	.	PAUSE_M

⊟ grid (default_ns)

MuH11MD::POS	PIAT	NN	VVINF	$,	ADV	VVFIN	PPER	PIS	VVINF	$.	PAUSE
MuH11MD::v	bisschen	fett	FRESsen		dann	gehn	wa	eine	RAUchen		(--)
seg::MuH11MD	bisschen	Fett	fressen	,	dann	gehen	wir	eine	rauchen	.	PAUSE_M

⊞ tree (MuH11MD_02)

9 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 74 - 84)

MuH11MD	PAUSE_M	Aber	dick	.	Ich	wollte	erst	UNINTERPRETABLE	#	Wer	denkst

⊟ grid (default_ns)

MuH11MD::POS	PAUSE	KON	ADJD	$.	PPER	VMFIN	ADV	XYU	$#	PWS	VVFIN
MuH11MD::v	(--)	aber	dick		ich	wollt	erst	(unverständlich)		wer	(de:nkst)
seg::MuH11MD	PAUSE_M	Aber	dick	.	Ich	wollte	erst	UNINTERPRETABLE	#	Wer	denkst

⊞ tree (MuH11MD_02)

2.2.6 Direkte Präzedenz

Wenn mehrere Sachen direkt nacheinander auftauchen sollen, können Sie dies mit der Zuweisung von Variablen und einem Punkt suchen. Wenn Sie beispielsweise alle Vorkommen finden möchten, in denen ein deskriptives Adjektiv direkt nach *zu* auftaucht, können Sie dies folgendermaßen finden:

```
/zu/ & POS=/ADJD/ & #1.#2
```

3 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 2110 - 2121) left context:

MuH11MD	immer	#	PAUSE_M	Bin	viel	zu	faul	,	um	aufzustehen	,	um

⊟ grid (default_ns)

MuH11MD::POS	ADV	$#	PAUSE	VAFIN	ADJD	PTKA	ADJD	$,	KOUI	VVIZU	$,	KOUI
MuH11MD::v	IMmer		(--)	bin	viel	zu	FAUL		um	AUFzustehen		um
seg::MuH11MD	immer	#	PAUSE_M	Bin	viel	zu	faul	,	um	aufzustehen	,	um

⊞ tree (MuH11MD_02)

2.2.7 Indirekte Präzedenz

Wenn mehrere Sachen irgendwann nacheinander auftauchen sollen, können Sie ähnlich vorgehen wie bei der direkten Präzedenz. Sie definieren Variablen und setzen sie mit .* in Relation zueinander. Wort 2 folgt auf Wort 1, wobei eine beliebige Menge an Elemente zwischen den beiden Wörtern stehen kann. Wenn Sie beispielsweise alle Vorkommen finden möchten, in denen ein Artikel irgendwann vor einem Nomen auftaucht, können Sie folgendermaßen suchen:

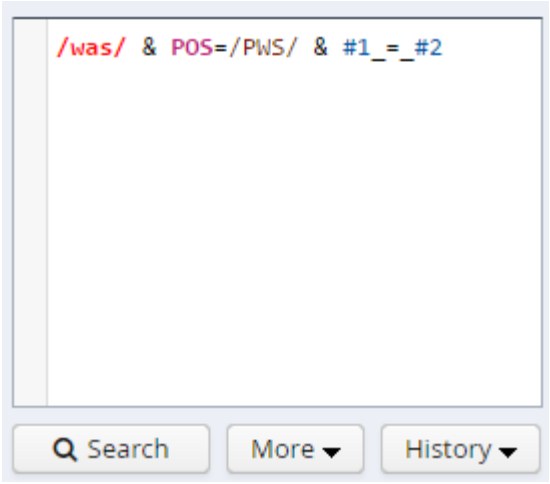
```
POS=/ART/ & POS=/NN/ & #1.*#2
```

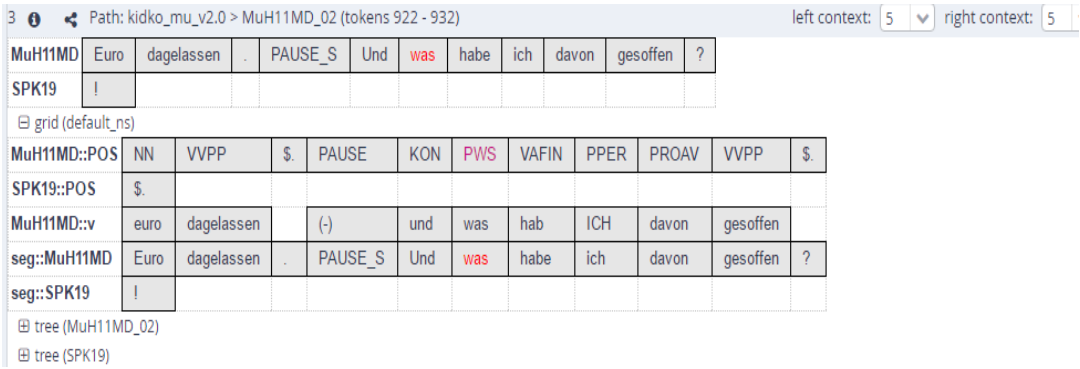
Achtung: ‚irgendwann' bedeutet tatsächlich ‚irgendwann'. Es passiert sehr oft, dass ein Artikel irgendwann vor einem Nomen steht, die beiden Elemente aber nicht zu derselben NP gehören (siehe Beispiel unten) oder sogar in zwei unterschiedlichen Sätzen auftauchen. Für eine Suche innerhalb eines Satzes ist TIGER besser geeignet (= s.o, Abschnitt zu TIGERSearch query language: In TIGER wird immer automatisch innerhalb von Sätzen gesucht, alle Merkmale, die dort in einer Anfrage verbunden werden, müssen also innerhalb eines Satzes zutreffen).

2.2.8 Abdeckung (X und Y treffen gleichzeitig zu)

Wenn Sie sagen wollen, das X sowohl Y als auch Z ist (zum Beispiel, wenn Sie „was" als Interrogativpronomen suchen, d.h. Sie suchen ein Wort, das sowohl das Lexem *was* als auch ein substituierendes Interrogativpronomen (PWS) sein soll), können Sie dies folgendermaßen suchen:

```
/was/ & POS=/PWS/ & #1_=_#2
```

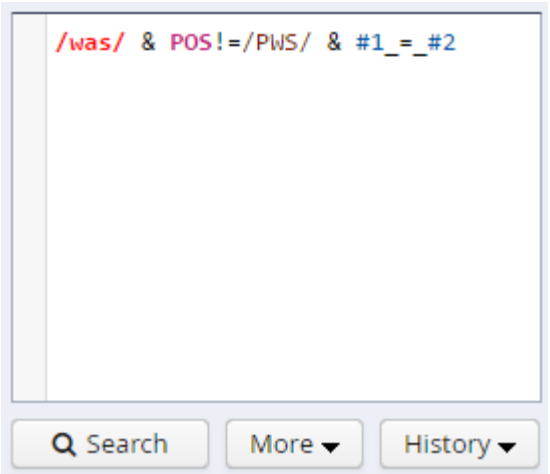




2.2.9 Negation

Bedingungen können auch negiert werden; ! bedeutet „nicht“. Wenn Sie z.B. alle Fälle von *was* suchen, die kein substituierendes Interrogativpronomen (PWS) sind, dann können Sie dies so erreichen:

```
/was/ & POS!=/PWS/ & #1_=_#2
```



2.2.10 Entwertung bestimmter Zeichen

Es könnte sein, dass Sie ab und zu ein Zeichen benutzen möchten, das schon eine Bedeutung in regulären Ausdrücken hat. Wenn Sie beispielsweise alle satzbeendenden Interpunktionen ($.) finden möchten, würden dieses Problem auftreten, weil sowohl $ als auch . schon eine Bedeutungen in regulären Ausdrücken haben. Wenn Sie also POS=/$./ eingeben würden, würden ein Fehlerhinweis auftauchen.

Um solche Zeichen trotzdem zu benutzen, müssen Sie einen umgekehrten Schrägstrich benutzen. Sie würden also eingeben:

```
POS=/\$\./
```

42 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 222 - 232)

SPK19	heißt	das	andere	?	Ah	!	Mann	!	Das	sieht	man

⊟ grid (default_ns)

SPK19::POS	VVFIN	ART	PIS	$.	SPITJ	$.	SPITJ	$.	PDS	VVFIN	PIS
SPK19::v	heißt	dis	andere		ah		mann		dis	sieht	man
seg::SPK19	heißt	das	andere	?	Ah	!	Mann	!	Das	sieht	man

⊞ tree (SPK19)

2.3 Strukturbäume in ANNIS

2.3.1 Einführung

Um syntaktische Bäume in ANNIS zu finden, nutzt man bekannten Suchanfragen in Annis (siehe Arbeiten mit ANNIS und ANNIS & reguläre Ausdrücke: Einführung). Zwei wichtige Unterschiede sind:

1) Sie können mit der Operation „>“ nach Dominanz suchen. (Siehe Beispiel 1 unten)
2) Um nach Ebenen in den syntaktischen Bäumen zu suchen, müssen Sie die Variable „cat“ (Category) benutzten. (Siehe Beispiel 1 unten).

Wie vorher, müssen zwei Punkte beachtet werden, wenn Sie eine *komplexe Suchanfrage* stellen wollen:

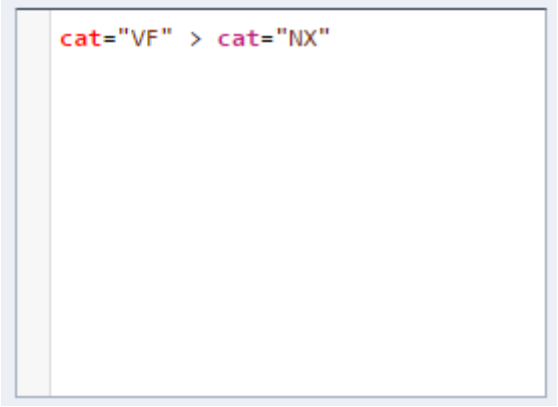
1) Es kann jeweils nur eine Operation („>“, „.“) zwischen zwei Elementen ausgedrückt werden.
2) Jedes Element muss durch eine *Variable* kodiert werden.

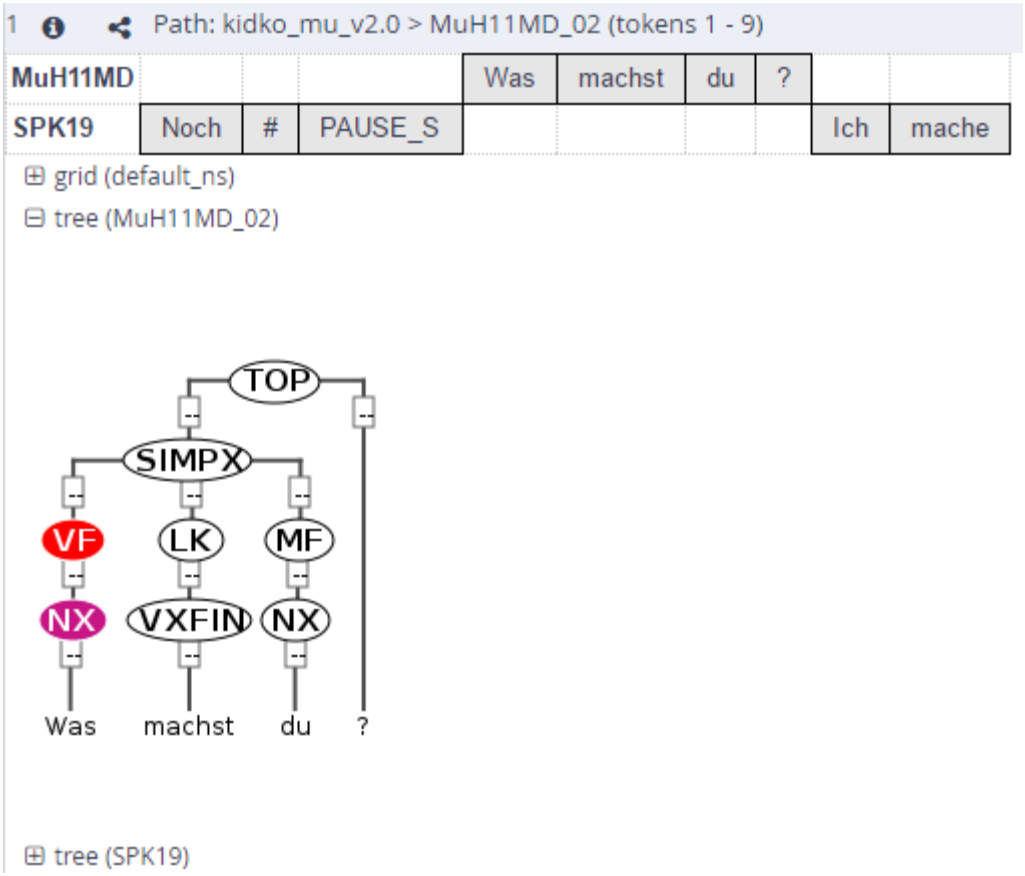
2.3.2 Beispiele

2.3.2.1 Beispiel 1: Nominale Element im Vorfeld

Sie wollen ein Vorfeld („VF“), das ein nominales Element, wie ein Nomen („NN“) oder einen Eigennamen („NE“), („NX“) direkt dominiert, suchen:

```
cat="VF" > cat="NX"
```

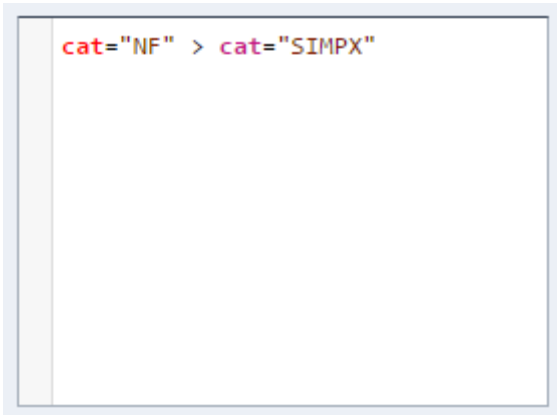


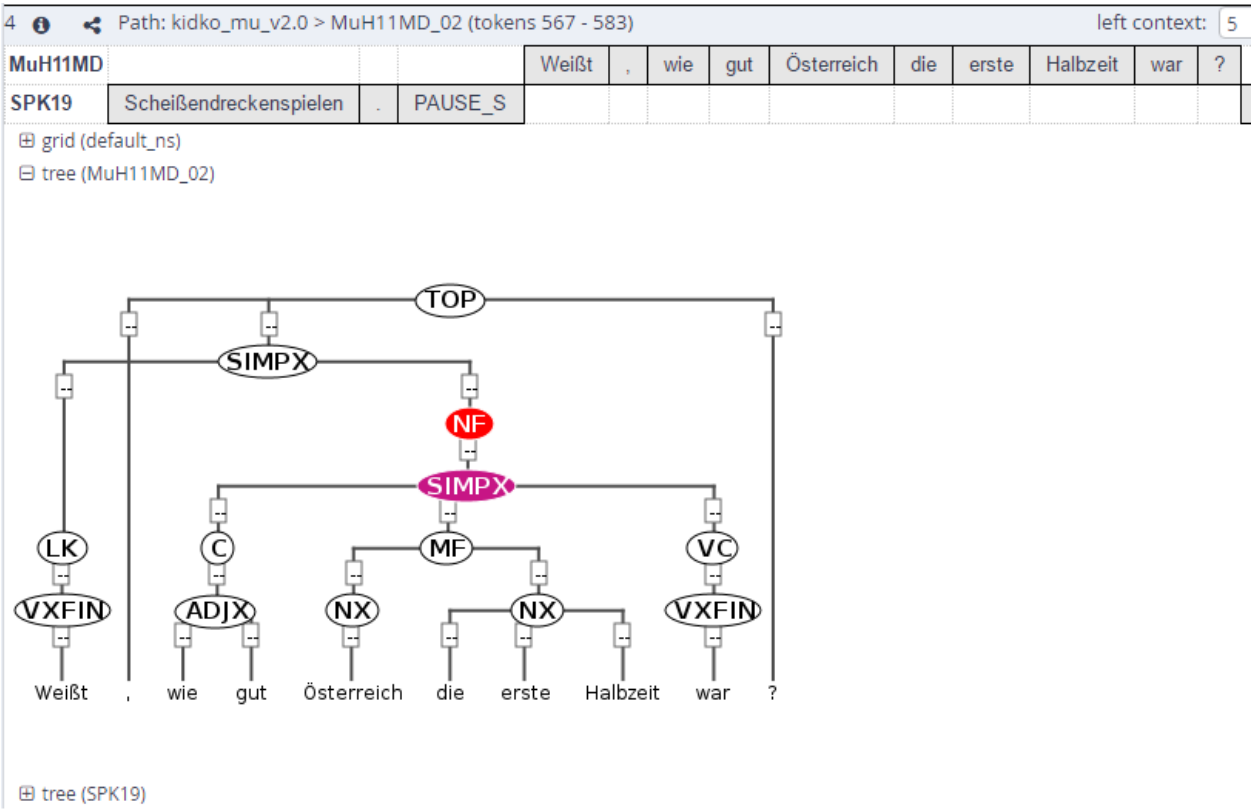


2.3.2.2 Beispiel 2: Nebensatz im Nachfeld

Sie suchen nach einem Nebensatz im Nachfeld:

```
cat="NF" > cat="SIMPX"
```



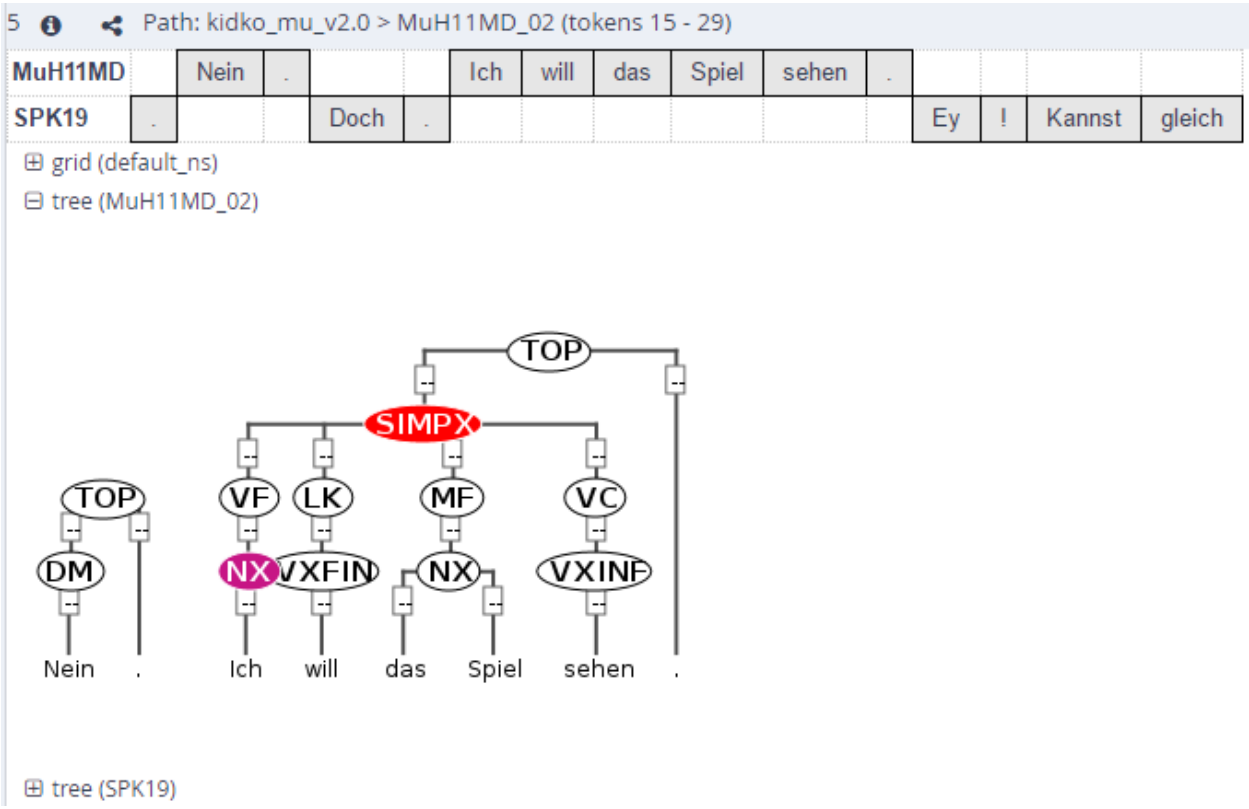


2.3.2.3 Beispiel 3: Nominales Element im Satz

Sie wollen einen Satzknoten, der ein NX beinhaltet, suchen. Der Asterisk bedeutet, dass *beliebig viele* Ebene dürfen zwischen der ersten Sache (hier, cat="SIMPX") und der zweiten Sache (hier, cat="NX") auftauchen können.

```
cat="SIMPX" >* cat="NX"
```

cat="SIMPX" >* cat="NX"



2.3.2.4 Beispiel 4: Nachfeld folgt indirekt nach einem Vorfeld

Sie wollen nach der Sequenz „Vorfeld („VF“) indirekt gefolgt von Nachfeld („NF“)“ suchen

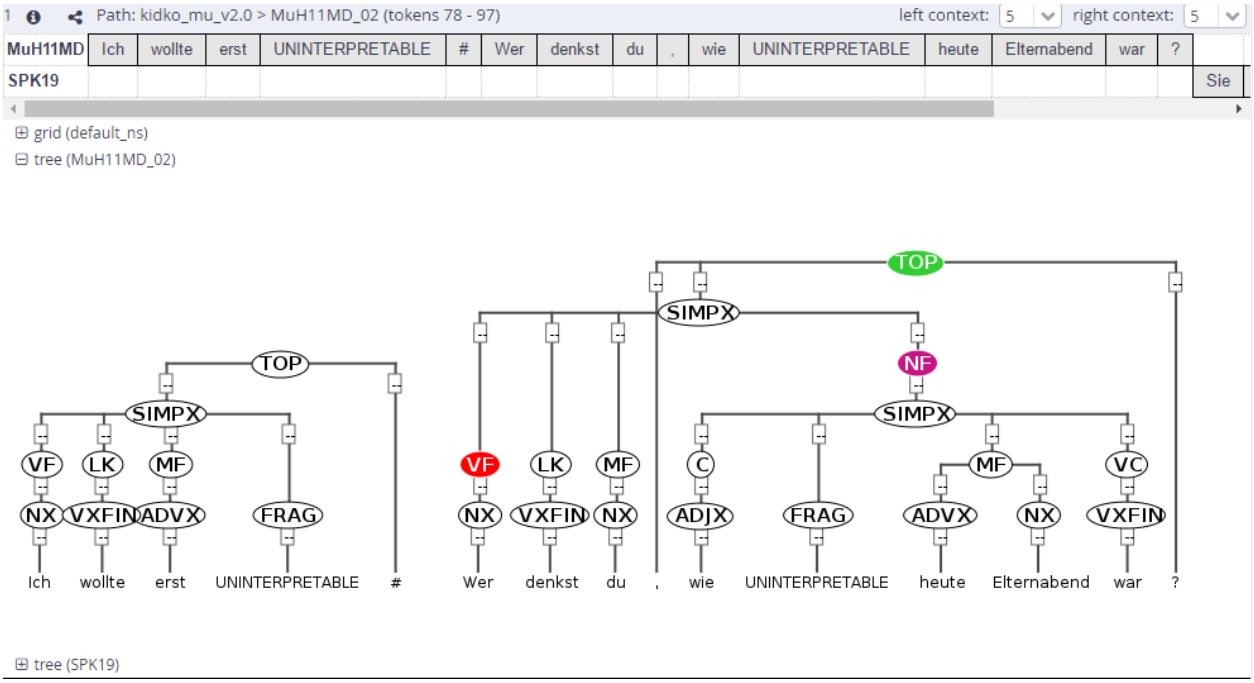
```
cat="VF" & cat="NF" & cat="TOP" & #1.*#2 & #3>*#1 & #3>*#2
```

Diese Suchanfrage sucht ein VF, auf das in der Struktur ein Nachfeld folgt (innerhalb eines Satzes). Das bedeutet: „Ich will alle Beispiele des Folgendes finden“:

- ein VF **und**
- ein NF **und**
- ein TOP **und**
- die erste Sache, die ich eingegeben habe (d.h. VF) ist indirekt von der zweiten Sache, die ich eingegeben habe (d.h. NF), gefolgt **und**
- die dritte Sache, die ich eingegeben habe (d.h. TOP) dominiert (indirekt) die erste Sache, die ich eingegeben habe (d.h. VF) (d.h. andere Ebene können zwischen diese zwei Ebene auftauchen) **und**
- die dritte Sache, die ich eingegeben habe (d.h. TOP) dominiert (indirekt) die zweite Sache, die ich eingegeben habe (d.h. VF).

Alle „**und**“s in der oberen Beschreibung korrespondieren mit den „&“s in der Suchanfrage und alle Farben in der oberen Beschreibung korrespondieren mit den Sachen der gleichen Farbe in der Suchanfrage.

```
cat="VF" &
cat="NF" &
cat="TOP" &
#1.*#2 &
#3>*#1 &
#3>*#2
```

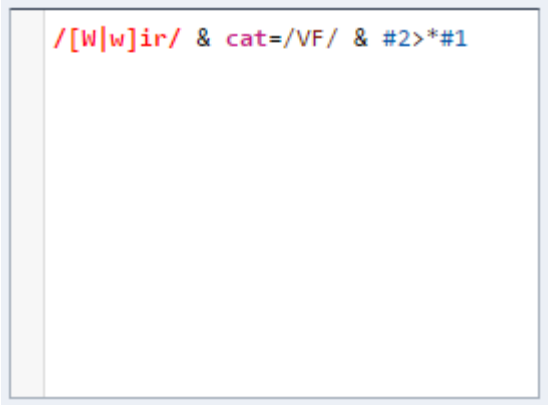


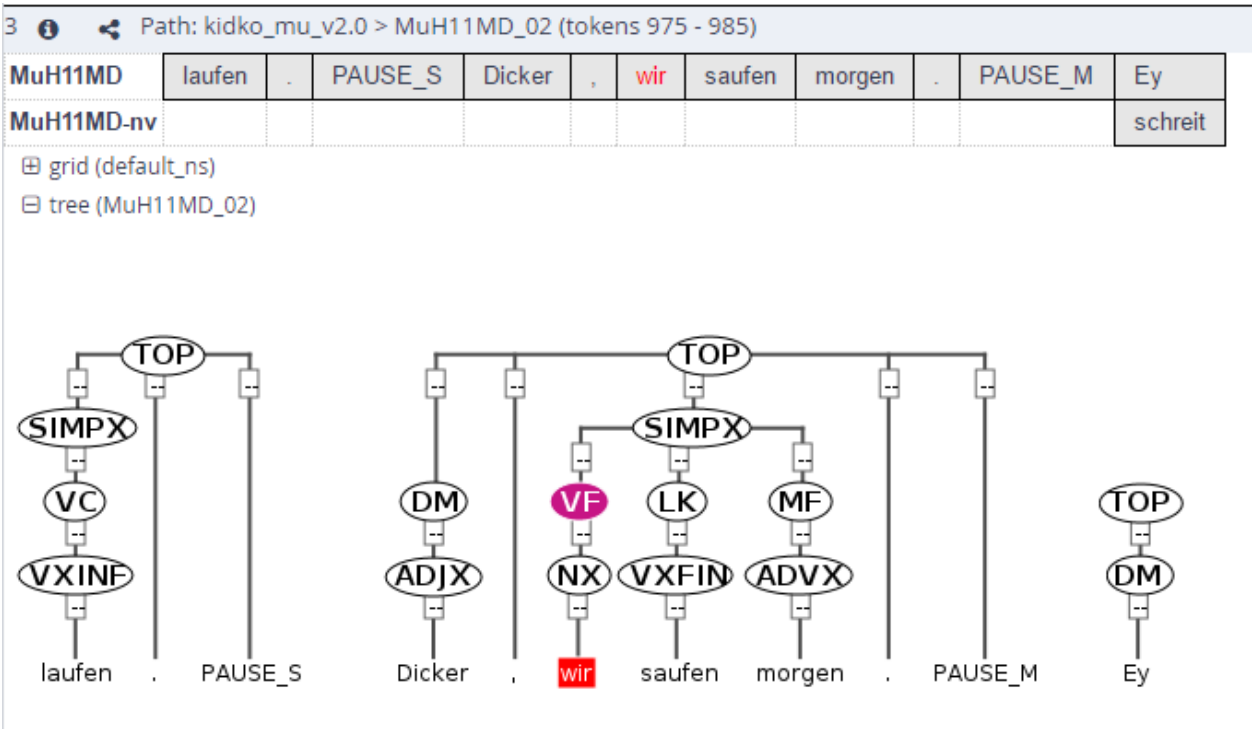
Bemerkung: Wenn Sie nur **cat="VF".*cat="NF"** suchen würden, wären die Treffer nicht auf *einen* einzigen Satz beschränkt. Das heißt, dass Sie auch Beispiele finden würden, in denen es ein VF in einem Satz und ein NF in dem nächsten Satz gibt. Die Begrenzung mit TOP legt fest, dass sowohl VF und NF im selben Satz auftauchen müssen.

2.3.2.5 Beispiel 5 : Syntaktische Bäume & (normalisierte) Wörter

Sie wollen alle Beispiele finden, in denen das (normalisierte) Wort *wir* (sowohl groß als auch klein geschrieben) im Vorfeld steht.

```
/[W|w]ir/ & cat=/VF/ & #2>*#1
```





2.3.2.6 Beispiel 6: Sequenz „Subjekt-finites Verb-Adverbialbestimmung“

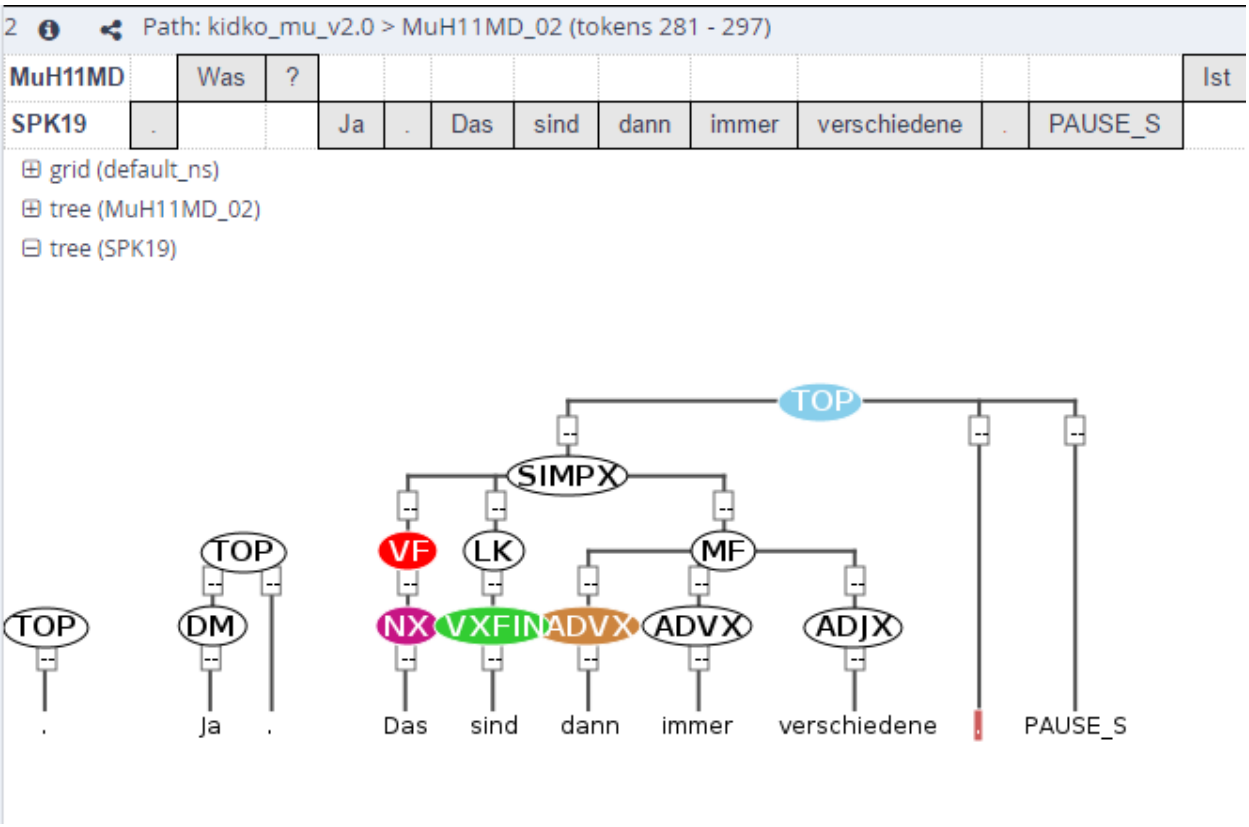
Sie wollen nach einem Deklarativsatz mit der Sequenz „Subjekt-finites Verb-Adverbialbestimmung“ suchen:

```
cat="VF" & cat="NX" & cat="VXFIN" & cat="ADVX" & cat="TOP"
&/\./ & #1>#2 & #2.#3 & #3.#4 & #5>*#1 & #5>*#3 & #5>*#4 &
#5>#6
```

Diese Suchanfrage kann so gelesen werden: „Ich will alle Beispiele finden, in denen:

- Es ein VF gibt **und**
- ein NX gibt **und**
- ein VXFIN gibt **und**
- ein ADVX gibt **und**
- ein TOP gibt **und**
- einen Punkt gibt **und**
- die erste Sache (d.h. VF) dominiert (direkt) die zweite Sache (d.h. NX) **und**
- die zweite Sache (d.h. NX) ist direkt von der dritten Sache (d.h. VXFIN) gefolgt **und**
- die dritte Sache (d.h. VXFIN) ist direkt von der vierten Sache (d.h. ADVX) gefolgt **und**
- die fünfte Sache (d.h. TOP) dominiert (indirekt) die erste Sache (d.h. VF) **und**
- die fünfte Sache (d.h. der gleichen TOP) dominiert (indirekt) die dritte Sache (d.h. VXFIN) **und**
- die fünfte Sache (d.h. der gleichen TOP) dominiert (indirekt) die vierte Sache (d.h. ADVX) **und**
- die fünfte Sache (d.h. der gleichen TOP) dominiert (indirekt) (d.h. beinhaltet) die sechste Sache (d.h. einen Punkt).“

```
cat="VF" &
cat="NX" &
cat="VXFIN" &
cat="ADVX" &
cat="TOP" &
/\./ &
#1>#2 & #2.#3 & #3.#4 &
#5>*#1 & #5>*#3 & #5>*#4 & #5>#6
```



2.4 Schnellübersicht

Eingabe	Beschreibung	Beispiele
`|`	Entweder / oder	`/zu[m|r]/` `/mach(|st|en|e|t)/`
`^`	Ausschließen von Zeichen	`/d[^ea]nn/`
`?`	Optionalität	`v=/jetz(t)?/`
`.`	Ein beliebiges Zeichen	`/lang../`
`.*`	Unbegrenzte Zahl an beliebige Zeichen	`POS=/AP.*/`
`#1.#2`	Direkte Präzedenz	`/zu/ & POS=/ADJD/ & #1.#2`
`#1.*#2`	Indirekte Präzedenz	`POS=/ART/ & POS=/NN/ & #1.*#2`
`#1_=_#2`	Abdeckung (X und Y treffen gleichzeitig zu)	`/was/ & POS=/PWS/ & #1_=_#2`
`!`	Negation	`POS!=/PWS/`
`\`	Entwertung bestimmter Zeichen	`POS=/\$\./`

2.5 ANNIS, KiDKo und Kiezdeutsch: Beispielsuchanfragen

2.5.1 Beispiel 1: Bloße NPs

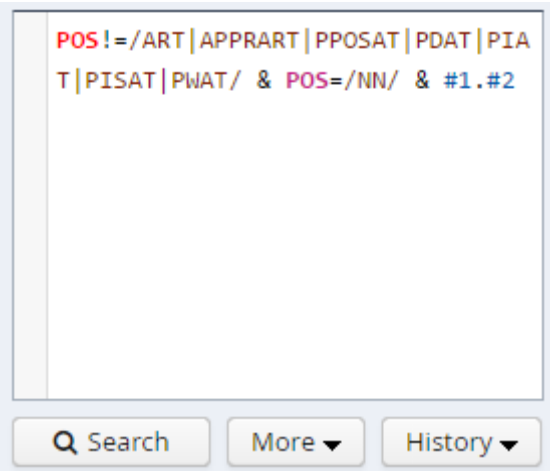
Stellen Sie sich folgende Situation vor: Sie haben gelesen, dass bloße NPs oft in Kiezdeutsch vorkommen, und Sie wollen herausfinden, was für bloße NPs in KiDKo zu finden sind. Sie könnten Folgendes suchen:

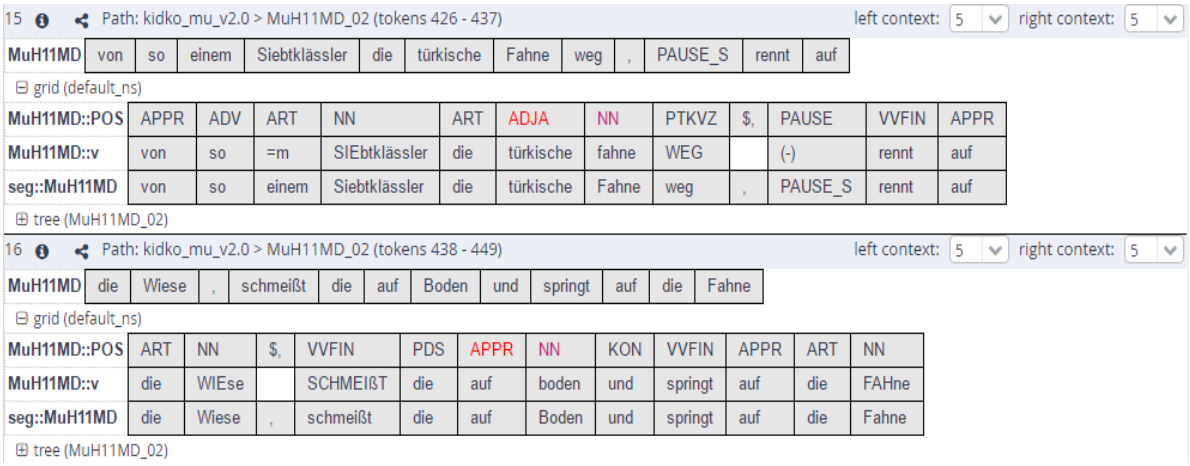
```
POS!=/ART|APPRART|PPOSAT|PDAT|PIAT|PISAT|PWAT/ & POS=/NN/
& #1.#2
```

Das heißt: Sie wollen alle Beispiele finden, in denen ein Nomen (NN) direkt nach etwas positioniert ist, das:

- kein (un)bestimmter Artikel (ART) (z.B. der, ein),
- keine Präposition mit Artikel (APPRART) (z.B. zum),
- kein attribuierendes Possessivpronomen (PPOSAT) (z.B. mein [Buch], deine [Mutter]),
- kein attribuierendes Demonstrativpronomen (PDAT) (z.B. jener [Mensch]),
- kein attribuierendes Indefinitpronomen ohne Determinier (PIAT) (z.B. kein [Mensch], irgendein [Glas]),
- kein attribuierendes Indefinitpronomen mit Determinier (PISAT) (z.B. [ein] wenig [Wasser], [die] beiden [Brüder]) oder
- kein attribuierendes Interrogativpronomen (PWAT) (z.B. welche [Farbe], wessen [Hut]) ist.

Vorsicht: Diese schließt Treffer wie „die türkische Fahne“ nicht aus. Die Funde müssen manuell geprüft oder die Suchanfrage weiter verfeinert werden.





15 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 426 - 437) left context: 5 right context: 5

MuH11MD	von	so	einem	Siebtklässler	die	türkische	Fahne	weg	,	PAUSE_S	rennt	auf

⊟ grid (default_ns)

MuH11MD::POS	APPR	ADV	ART	NN	ART	ADJA	NN	PTKVZ	$,	PAUSE	VVFIN	APPR
MuH11MD::v	von	so	=m	SIEbtklässler	die	türkische	fahne	WEG		(-)	rennt	auf
seg::MuH11MD	von	so	einem	Siebtklässler	die	türkische	Fahne	weg	,	PAUSE_S	rennt	auf

⊞ tree (MuH11MD_02)

16 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 438 - 449) left context: 5 right context: 5

MuH11MD	die	Wiese	,	schmeißt	die	auf	Boden	und	springt	auf	die	Fahne

⊟ grid (default_ns)

MuH11MD::POS	ART	NN	$,	VVFIN	PDS	APPR	NN	KON	VVFIN	APPR	ART	NN
MuH11MD::v	die	WIEse		SCHMEIßT	die	auf	boden	und	springt	auf	die	FAHne
seg::MuH11MD	die	Wiese	,	schmeißt	die	auf	Boden	und	springt	auf	die	Fahne

⊞ tree (MuH11MD_02)

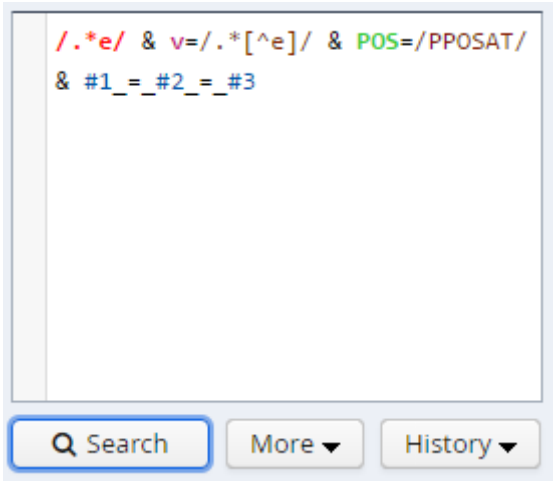
2.5.2 Beispiel 2: Possessive Adjektivendungen

Stellen Sie sich vor, sie haben gehört, dass possessive Adjektivendungen in Kiezdeutsch mitunter fehlen (z.B. meine Schwester → mein Schwester). Sie möchten nun herausfinden, wie oft diese Strukturen auftauchen. Dazu könnten Sie Folgendes eingeben:

```
/.*e/ & v=/.*[^e]/ & POS=/PPOSAT/ & #1_=_#2_=_#3
```

Das heißt: Sie wollen alle attribuierenden Possessivpronomen (PPOSAT) finden, die in der normalisierten Ebene auf ein *-e* enden, aber nicht in der verbalen Ebene.

Vorsicht: Wie Sie unten sehen können, sind die Trefferpaare nicht auf einen bestimmten Sprecher begrenzt, Sie erhalten also auch ein paar „false positives“, d.h. Treffer, bei denen ein Sprecher z.B. „mein“ gesagt hat, während ein anderer Sprecher gleichzeitig etwas gesagt hat, für das in der Normalisierungsebene ein PPOSAT steht, das auf *-e* endet.



2 Path: kidko_mu_v2.0 > MuH11MD_04 (tokens 6189 - 6199) left c

MuH11MD				Das	ist	meine	erste	Freundin	,	die	ich			
SPK101	ja	geil	.											

⊟ grid (default_ns)

MuH11MD::POS				PDS	VAFIN	PPOSAT	ADJA	NN	$,	PRELS	PPER			
SPK101::POS	ADV	ADJD	$.											
MuH11MD::v				dis	is	mei	ERSte	freundin		die	ich			
SPK101::v	ja	GEIL												
seg::MuH11MD				Das	ist	meine	erste	Freundin	,	die	ich			
seg::SPK101	ja	geil	.											

⊞ tree (MuH11MD_04)

⊞ tree (SPK101)

3 Path: kidko_mu_v2.0 > MuH11MD_06-2 (tokens 264 - 274) left c

MuH11MD			Das	wäre	schlimm	ge	#							
SPK22	schon	.			Weil	mein	Bein	war	so	und	du			

⊟ grid (default_ns)

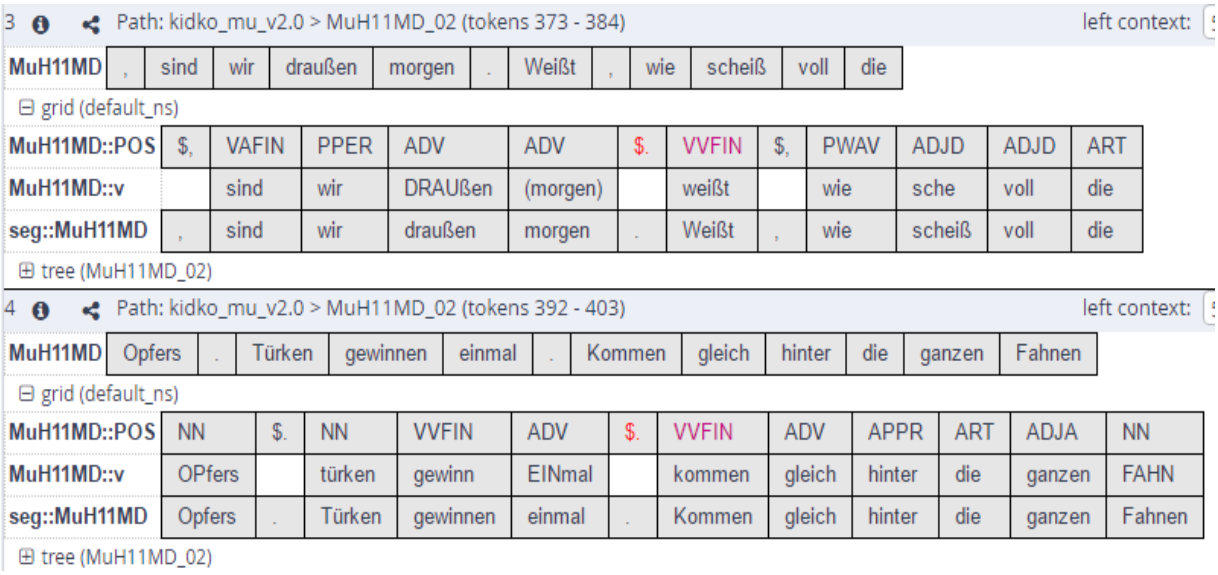
MuH11MD::POS			PDS	VAFIN	ADJD	XYB	$#							
SPK22::POS	ADV	$.			KON	PPOSAT	NN	VAFIN	ADV	KON	PPER			
MuH11MD::v			dis	wär	SCHLIMM	ge								
SPK22::v	schon				weil	mein	bein	war	SO	und	du			
seg::MuH11MD			Das	wäre	schlimm	ge	#							
seg::SPK22	schon	.			Weil	mein	Bein	war	so	und	du			

⊞ tree (MuH11MD_06-2)

2.5.3 Beispiel 3: Finites Verb am Satzanfang

Stellen Sie sich vor, sie haben gehört, dass finite Verben auch im Satzanfang stehen können (auch, wenn es keine „ja/nein“-Frage oder ein Imperativsatz ist), und Sie wollten dieses Phänomen in Kiezdeutsch genauer untersuchen. Dazu könnten Sie Folgendes eingeben:

```
POS=/\$\./ & POS=/.*FIN/ & #1.#2
```



3 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 373 - 384) left context:

MuH11MD	,	sind	wir	draußen	morgen	.	Weißt	,	wie	scheiß	voll	die
grid (default_ns)												
MuH11MD::POS	$,	VAFIN	PPER	ADV	ADV	$.	VVFIN	$,	PWAV	ADJD	ADJD	ART
MuH11MD::v		sind	wir	DRAUßen	(morgen)		weißt		wie	sche	voll	die
seg::MuH11MD	,	sind	wir	draußen	morgen	.	Weißt	,	wie	scheiß	voll	die
tree (MuH11MD_02)												

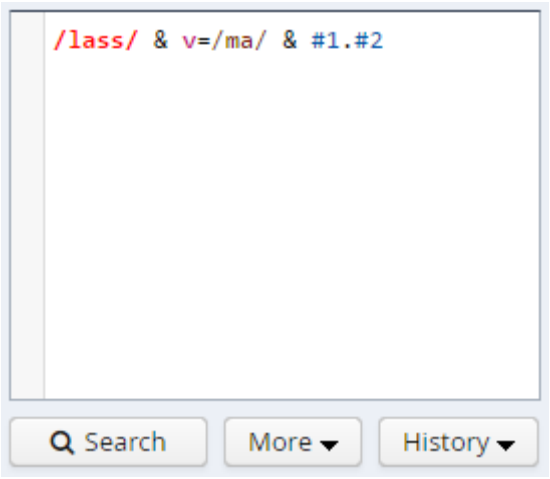
4 Path: kidko_mu_v2.0 > MuH11MD_02 (tokens 392 - 403) left context:

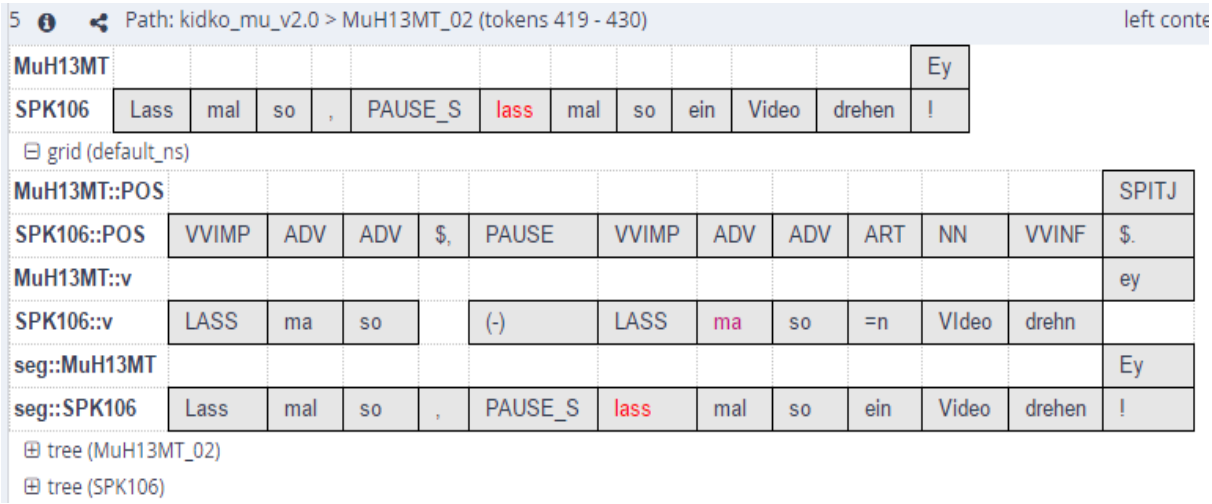
MuH11MD	Opfers	.	Türken	gewinnen	einmal	.	Kommen	gleich	hinter	die	ganzen	Fahnen
grid (default_ns)												
MuH11MD::POS	NN	$.	NN	VVFIN	ADV	$.	VVFIN	ADV	APPR	ART	ADJA	NN
MuH11MD::v	OPfers		türken	gewinn	EINmal		kommen	gleich	hinter	die	ganzen	FAHN
seg::MuH11MD	Opfers	.	Türken	gewinnen	einmal	.	Kommen	gleich	hinter	die	ganzen	Fahnen
tree (MuH11MD_02)												

2.5.4 Beispiel 4: Lassma

Vielleicht haben Sie gehört, dass Kiezdeutsch SprecherInnen oft ‚lass ma(l)' (lass uns mal) sagen und Sie wollen die Kontexte genauer angucken, in denen ‚lassma' auftaucht. Dazu könnten Sie Folgendes eingeben:

```
/lass/ & v=/ma/ & #1.#2
```



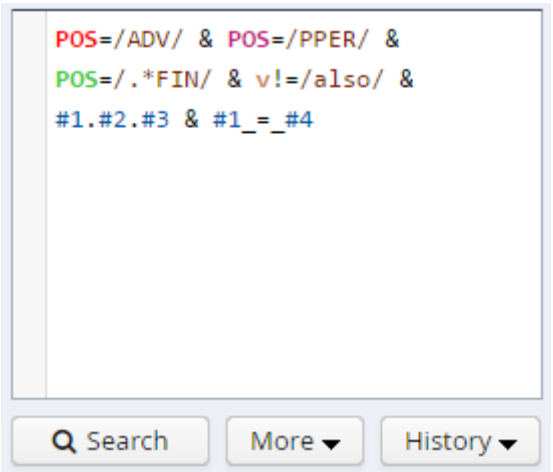


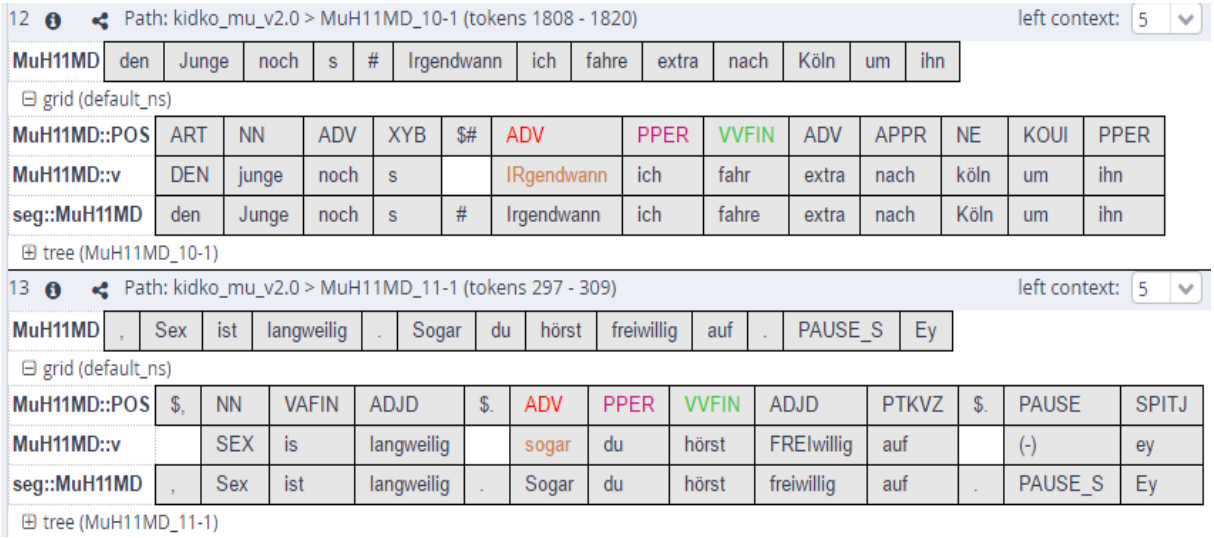
2.5.5 Beispiel 5: Nichtkanonische Adverbialbestimmung-Subjekt-Verb (fin) – Strukturen

Stellen Sie sich vor, sie haben gehört, dass Adverbialbestimmung-Subjekt-Verb (fin) - Strukturen in Kiezdeutsch auftauchen (z.B. „Nachher wir gehen ins Kino.“). Sie möchten nun diese Struktur genauer untersuchen. Dazu könnten Sie Folgendes eingeben:

```
POS=/ADV/ & POS=/PPER/ & POS=/.*FIN/ & v!=/also/ & #1.#2.#3
& #1_#4
```

Mit der Bedingung `v!=/also/` eliminiert Sie viele Treffer, da Sie nicht die standardsprachlichen, kanonischen Strukturen, wie „Also wir gehen ins Kino“, interessieren.





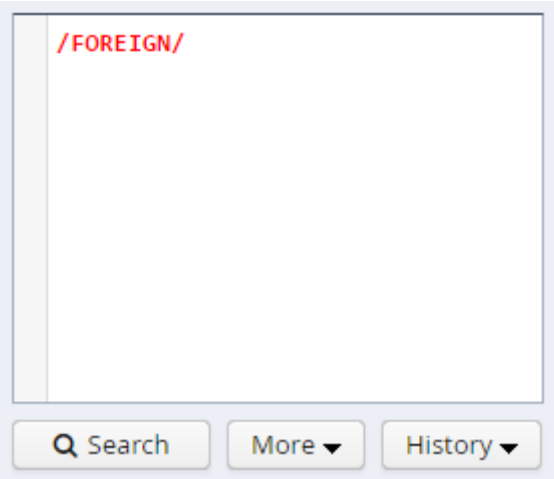
Auch diese Suche findet Sätze, in denen keine nichtkanonische Adverbialbestimmung – Subjekt – Verb (fin) – Struktur auftaucht (siehe unteres Beispiel). Die Treffer müssen also manuell überprüft oder die Suchanfrage weiter verfeinert werden.

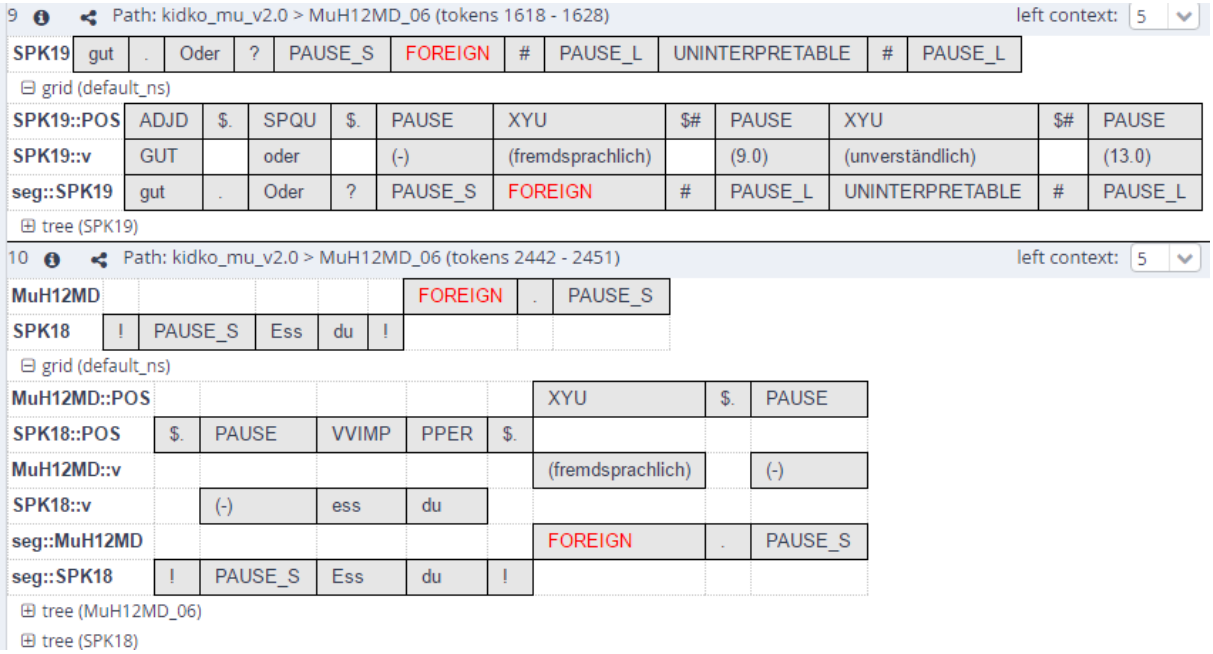
2.5.6 Beispiel 6: Code-Switchung

Vielleicht interessiert es Sie, wie Code-Switching funktioniert, also in welchen Kontexten oder mit welchen grammatischen oder lexikalischen Einheiten dieses Phänomen auftritt. Im KiDKo wurden alle fremdsprachlichen Gesprächsbeiträge getaggt (vgl. Tags im Kiezdeutschkorpus) und durch FOREIGN ersetzt.

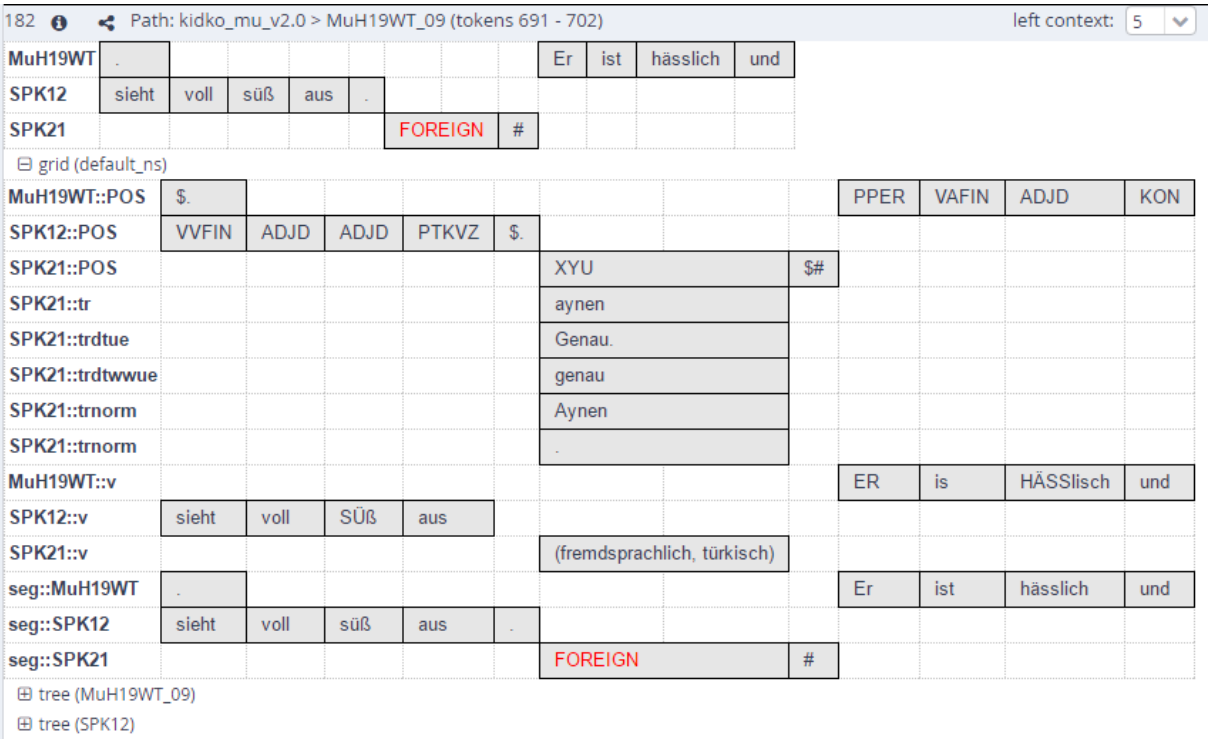
Sie könnten daher folgendermaßen suchen:

/FOREIGN/





Für einige fremdsprachliche Gesprächsbeiträge bietet das Korpus eine Übersetzung an.


182
Path: kidko_mu_v2.0 > MuH19WT_09 (tokens 691 - 702)
left context: 5
MuH19WT . Er ist hässlich und
SPK12 sieht voll süß aus .
SPK21 FOREIGN #
grid (default_ns)
MuH19WT::POS $. PPER VAFIN ADJD KON
SPK12::POS VVFIN ADJD ADJD PTKVZ $.
SPK21::POS XYU $#
SPK21::tr aynen
SPK21::trdtue Genau.
SPK21::trdtwwue genau
SPK21::trnorm Aynen
SPK21::trnorm .
MuH19WT::v ER is HÄSSlisch und
SPK12::v sieht voll SÜß aus
SPK21::v (fremdsprachlich, türkisch)
seg::MuH19WT . Er ist hässlich und
seg::SPK12 sieht voll süß aus .
seg::SPK21 FOREIGN #
tree (MuH19WT_09)
tree (SPK12)