SEARCH MANUAL

for

VOICE 3.0 Online

When referring to this document, please cite:

Osimk-Teasdale, Ruth; Pirker, Hannes; Pitzl, Marie-Luise. 2021. *Search manual for VOICE 3.0 Online.* https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/09/Search-manual-VOICE-3.0-Online.pdf (date of last access).

Contents

INTRODUCTION 1

PART I GENERAL PRINCIPLES 2

1. QUERIES IN VOICE 3.0 ONLINE: OVERVIEW 2

1.1. Searching for word forms (token queries) 2

1.2. POS queries 2

1.3. Lemma queries 2

1.4. Mark-up search - NEW!!! 3

1.5. Phrase search 3

2. FINE-TUNING SEARCHES 4

2.1. Wildcards 4

2.2. Boolean operator AND: , (comma) 4

2.3. Boolean operators OR: | (vertical line) 4

2.4. Searching "within" mark-up 4

2.5. Searching for mark-up "containing" 5

2.6. Number-controlled placeholders for random words 5

3. BACKGROUND INFORMATION 5

3.1. Technical background 5

3.2. Regular Expressions - Searching with wildcards and more 6

PART II EXAMPLES 10

4. TOKEN SEARCH (WORD FORMS) 10

4.1. Simple token search 10

4.2. Token search with wildcards 10

5. POS AND LEMMA SEARCH 11

5.1. Simple POS search 11

5.2. POS search with wildcards 11

5.3. Lemma search 12

6. MARK-UP SEARCH - NEW!!! 12

6.1. Searches for tokenised mark-up 12
6.1.1. Pauses 12
6.1.2. Laughter 13

6.2. Searches for mark-up in pointed brackets 13
6.2.1. Speaking modes 13
6.2.2. Non-English Speech 14
6.2.3. Overlaps 15
6.2.4. Onomatopoeia 15
6.2.5. Speaker noises 15
6.2.6. Utterances 15

6.3. Searches for mark-up via POS tags or token prefix 16

7. PHRASE SEARCH – PARTLY NEW!!! 17

7.1. Lexical phrases (tokens) 17

7.2. Part-of-speech and lemma combinations 17

7.3. Token, POS, lemma combinations 17

7.4. Token and mark-up sequences 18

7.5. POS and mark-up sequences 19

8. EXPERT SEARCH – NEW!!! 19

8.1. Fine-tuning searches (and) 19

8.2. Fine-tuning searches (or) 20

8.3. Using placeholders in phrase search 22

8.4. Search *within*: Find tokens and POS/lemmas within mark-up 23
8.4.1. Tokens within Mark-up 23
8.4.2. POS within Mark-up 23
8.4.3. Lemma within Mark-up 23

8.5. Search *containing*: Find stretches of speech with particular mark-up that contain particular tokens/POS/lemmas 24
8.5.1. Mark-up containing token 24
8.5.2. Mark-up containing POS 24
8.5.3. Mark-up containing lemma 24

8.6. Combined searches with wildcards and fine-tuning 25

8.7. Combined searches with *within* or *containing* 27

8.8. Combined searches with placeholder 28

8.9. Combined mark-up searches 28

9. LINKS 29

INTRODUCTION

This search manual seeks to provide a guide to performing queries and using query language in VOICE 3.0 Online. VOICE 3.0 Online is an updated and enhanced version of the *Vienna-Oxford International Corpus of English*. It replaces previous online versions (VOICE 1.0, 1.1., 2.0) of the corpus.

VOICE 3.0 Online was developed within the interdisciplinary digital humanities project VOICE CLARIAH between April 2020 and September 2021 with the aim to ensure long-term online availability of the corpus. Within the project, the system architecture behind VOICE Online was updated and an entirely new backend was developed and implemented. The previously separate VOICE XML and VOICE POS XML versions were merged and a completely new frontend (i.e. web interface) for VOICE Online was designed. This new frontend is called VOICE 3.0 Online. It was released as an open-access interface in September 2021 and offers a variety of new features for corpus users. The search syntax, which has been considerably expanded and partly changed from previous versions of VOICE, is outlined in this manual.

The manual is split into two parts.

PART I General principles explains the principles underlying searches in VOICE 3.0 Online. Section 1 provides a brief overview of different types of searches that are possible in VOICE 3.0 Online. Section 2 provides a summary for fine-tuning searches in VOICE 3.0 Online. Section 3 offers background information and explains details of how regular expressions can be used in VOICE 3.0 Online. This section might be of interest to more long-term and/or experienced users of VOICE.

PART II Examples is primarily organized into tables. Sections 4 to 8 provide a substantial collection of examples for different types of queries with explanations and search results for users to draw on.

Links to additional corpus information and detailed documentation for VOICE Online and VOICE XML are provided in Section 9: Links.

Part I General principles

1. Queries in VOICE 3.0 Online: Overview

1.1. Searching for word forms (token queries)

- What we refer to as token queries are, generally speaking, searches for orthographic words or word forms separated by a space in the transcript.
- **Token query**: **enter the token using lower-case characters, e.g. *speak.***
- All queries are **case-sensitive**. Tokens are searched for with **lower case characters**, e.g., *i speak french.* Capital letters indicate a POS search (see section 1.2. POS queries).
- **Contracted forms** (e.g. *wanna, gonna, don't, it's*) need to be searched for with a space inserted before the contracted part, i.e. *wan na, gon na, do n't, it 's*
- **For further examples**, see 4.1. Simple token search.

1.2. POS queries

- POS queries allow searching for the Part-of-Speech annotations (i.e., morphosyntactic categories) of tokens.
- **POS search**: **enter the POS tag in capital letters, e.g., *VVP*.**
- All tokens in VOICE are POS-tagged in a double annotation scheme. This means that each token has been annotated with an individual POS tag for morphological form and, in parentheses, a POS tag for syntactic function. These are often, though not always, identical, as in *professional_JJ(JJ)*.
- If a POS tag is searched for without further specification in VOICE 3.0 Online, both positions (i.e. form and function) are searched.
- To search either form or function position separately, enter p:POS for form position or f:POS for function position.
- For information on the POS tagging of VOICE and the POS tagset see VOICE Tagset and POS tagging Manual.
- **For further examples,** see 5.1. Simple POS search.

1.3. Lemma queries

- A lemma is the basic form of a word, which represents all declensions and inflected forms of a word, e.g. *walk* is the lemma of *walk, walks, walked, walking*.
- **Lemma search: *l:lemma*, e.g. *l:walk***
- For details see also 5.3. Lemma search.

1.4. Mark-up search - NEW!!!

- Conversational mark-up can be searched for and retrieved in different ways in VOICE 3.0 Online. Users can
    - search for tokenized mark-up,
    - retrieve words with particular mark-up categories through designated token prefixes
    - or search POS tags indicating mark-up.
    - In addition, for the first time in VOICE 3.0 Online, users can also search for words, POS and lemmas that occur within and between stretches of conversational mark-up in the corpus.
- Mark-up search via **tokenized mark-up**: e.g. pauses (e.g. *_1*, *_2*, etc.) and laughter (e.g. _@, _@@@), see 6.1. Searches for tokenised mark-up.
- Mark-up search via designated **token prefixes**: e.g. *ƒ_* for foreign words, *s_* for spelled words, see 6.3. Searching for mark-up via POS tags or token prefixes.
- Mark-up search via designated **POS tags**: e.g. *PVC,* see see 6.3. Searching for mark-up via POS tags or token prefixes.
- Mark-up search for and between pointed brackets, i.e. retrieving **stretches of conversational mark-up**, e.g. *<soft/>, <L1ita/>, *. This type of query makes stretches of **speaking modes, non-English speech,** or **overlapping speech (
)** between pointed brackets searchable, see 6.2. Searches for mark-up in pointed brackets.
- For descriptions of the mark-up categories used for transcribing VOICE, see the VOICE Mark-up Conventions.
- Detailed examples for all mark-up searches are provided in section 6. Mark-up search.

1.5. Phrase search

- Any combination of token, POS, lemmas, searchable mark-up and placeholders for random words can be searched for as well as combined into phrases in any order with each item separated by a space.
- **Phrase search**: e.g. *the JJ l:university*.
- N.B. Phrase searches are only carried out **within individual utterances**. Therefore, phrases that go beyond utterance boundaries will not be found.
- **Conversational mark-up** such as pauses, laughter, breathing, tags for overlapping speech and other mark-up are **ignored** in phrase search (i.e. they do not break up lexical phrases), unless they are explicitly included in the (phrase) search via Mark-up search.
- For further examples, see 7. PHRASE SEARCH.

2. Fine-tuning searches

2.1. Wildcards

- VOICE 3.0 uses regular expressions (regex) for writing queries which employ wildcards. For a detailed explanation of this feature, see 3.2. Regular Expressions - Searching with wildcards and more
- Note to users of previous versions of VOICE Online:
  Because VOICE 3.0 Online uses regex, the syntax of wildcard search has changed in VOICE 3.0 Online (compared to VOICE 2.0 Online). As a general rule, you now need to insert a full stop before any wildcard character to obtain the search results you are familiar with from previous versions of VOICE Online, e.g. *.*, .+ or .?*
- For examples see 4.2. Token search with wildcards, 5.2. POS search with wildcards.

2.2. Boolean operator AND: , (comma)

- The comma serves as the Boolean operator AND in the query syntax used in VOICE 3.0 Online.
- A query for *condition1**,**condition2* finds items that fulfil the condition1 AND condition2. Note that there is no space: *condition1,condition2*.
- Typical usage: Finds **sub-specifications** of tokens with POS tags or lemmas. Any sequence of tokens, lemmas or POS tags before and after the comma is possible, e.g. *flat,JJ* or *VV,get* or *like,DM* etc.
- For more examples see 8.1. Fine-tuning searches (and).

2.3. Boolean operators OR: | (vertical line)

- The vertical line | serves as the Boolean operator OR in the query syntax used in VOICE 3.0 Online.
- Searching for *condition1*|*condition2* finds items that fulfil condition1 OR condition2. Any sequence of tokens, lemmas or POS tags before and after the vertical line is possible, e.g. *RB*|*JJ good*, *yeah* _@|*PA* etc.
- For more examples see 8.2. Fine-tuning searches (or).

2.4. Searching "within" mark-up

- Searching for a word form, POS tag or lemma (or any combination thereof) ***within*** *<mark-up/>* finds and highlights the searched word form, POS, lemma or phrase only if it is located somewhere within the specified <mark-up/> tag (in pointed brackets).
- The 'operator' *within* performs this type of specialized query, e.g. *well within , great within <soft/>.*

- For more examples see 8.4. Search *within*
- NB. The word 'within' only takes on this special function as an 'operator' in queries that make use of <mark-up/> tags. In regular token queries, the word 'within' will be treated like any other word or word form.

2.5. Searching for mark-up "containing"

- VOICE 3.0 Online also allows you to search for stretches of *<mark-up/>* ***containing*** particular word forms, POS tags, lemmas or phrases.
- The 'operator' *containing* performs this type of specialized query, e.g. * containing well, <soft/> containing great.*
- For examples see 8.5. Search *containing*
- NB. The word 'containing' only takes on this special function as an 'operator' in queries that make use of <mark-up/> tags. In regular token queries, the word 'containing' will be treated for like any other word or word form.

2.6. Number-controlled placeholders for random words

- .* can be used as a *placeholder* for any word in a phrase search.
- To specify that you would like to look for e.g. three random words in a row you can use .* multiple times, separated by spaces.
- Please note that this use is only meaningful if combined with other lexical tokens, POS or mark-up tags before and/or after .*
- A more concise way to formulate the same query is to use number-controlled placeholders using the {...} notation.
- {number}: specifies an exact number of random words you are looking for, e.g. *{1} well* retrieves *ah well, oh well, er well, yes well,* etc. (so *well* preceded by 1 left collocate).
- {minimal number,maximal number}: specifies a range of minimal and maximal number of random words that co-occur with the search item, e.g. *yeah {1,3} well* retrieves *yeah as well, yeah actually maybe well, yeah this is great well*.
- Examples: 8.3. Using placeholders in phrase search.

3. Background information

3.1. Technical background

Searches in VOICE 3.0 Online are handled by a tool called NoSketchEngine (NoSke) (https://nlp.fi.muni.cz/trac/noske), which is a powerful and efficient corpus-query engine.

CQL is the query language used by the NoSke (see https://www.sketchengine.eu/documentation/cql-basics/). Although CQL is very powerful and expressive, it is rather verbose. For the application and development of VOICE 3.0 Online, we therefore decided to provide an alternative, simplified query syntax (described in this manual), which users of VOICE enter into the frontend (i.e. the corpus interface of VOICE 3.0). The queries entered by users are automatically translated into CQL and executed by a NoSke instance.

Having entered a query in VOICE 3.0 Online, you can actually see the resulting 'translation' of your query in CQL displayed below the search field. If you would like to report unexpected search behavior to us, we recommend that you also provide the displayed CQL syntax of your query (e.g. include this in an email to us).

Alternatively, VOICE 3.0 Online also allows you to use CQL directly. Feel free to use CQL in the frontend if this is your preferred query language.

3.2. Regular Expressions - Searching with wildcards and more

In Part II of this manual, we rely on an extensive collection of examples to acquaint you with the query syntax used in VOICE 3.0 Online.

Yet, because regular expressions — or in short *regex* — are a fundamental concept, which is used in different contexts, they deserve a more elaborate introduction in this manual.

Regular expressions are a widely used way to define search patterns. The possibility to make use of regexes is one of the core capabilities of the search engine built for VOICE 3.0 Online.

The next subsections therefor present a summary of the main elements and syntax of regex.

3.2.1. *Placeholder for character*

. Matches any single character. You can perceive this as a kind of universal 'joker'.

Examples: m.n -> man, men
hi. -> him, his, hit
h.t -> hat, hit, hot, hut
m..n -> mean, main

[...] Character class: Matches any character contained in the bracket

[^...] *Inverted* character class: Matches any character *not* contained in the bracket

Examples: [hc]at -> hat, cat
h[ai]t -> hat, hit
h[^ai] -> hot, hut

3.2.2. Quantifiers

? The preceding element can appear *0 or 1 times*, i.e. is *optional*

Examples: houses? -> house, houses
fill?ing -> filing, filling

+ The preceding element must appear *1 or more* times, i.e. it is *not* optional and might be repeated.

Example: house.+ -> houses, household, housewives
(i.e. all words that start with 'house' plus at least one more character)

***** The preceding element can appear *0 or more* times, i.e. it is optional and might be repeated.

Note to users of previous versions of VOICE Online:

You might be familiar with the usage of plain ? + * from previous versions of VOICE Online or from other tools. Such systems do not use regexes but a search syntax called wildcards. The difference looks small but is significant.

In wildcard syntax "*" already denotes "zero or more characters", while in regex "*" is a quantifier which operates on the *preceding* element. Therefore, wildcard house* will find house, houses, household etc. In regex, the use of "*" in house* will only quantify the final "e", and thus will only match hous, house, housee, houseeee, etc.

If you have worked with previous versions of VOICE Online (and thus may be accustomed to search behaviour that uses wildcard syntax), remember to use the additional placeholder character ". " (see section 3.2.1) in VOICE 3.0 Online immediately before typing in any of the quantifiers (i.e. search for *house.*, house.?, house.*+).

Examples: house.* -> house, houses, household, housewives
(i.e. all words that start with "house" plus 0 or more characters)
.*ize -> organize, apologize, harmonize, ...

To gain even more precise control over the number of allowed and necessary character repetitions, you can use:

{min,max} The preceding element must appear at least **min** and not more than **max** times.

{min,} An empty **max** means there is no upper limit.

{0,1} is equivalent to ?
{1,} is equivalent to +
{0,} is equivalent to *

3.2.3. Boolean OR and Grouping

Boolean OR

| Indicates alternatives: matches either the left or the right of |.

Example: this|that -> matches this, that
he|she|it -> matches he, she, it

Grouping

(…)Brackets can be used to group characters (and even regular expressions) to form new elements. When describing the quantifiers above, we have seen that they operate on preceding elements. Up to now, these preceding elements always were just single letters. By using brackets in the search syntax, we can, however, group multiple characters (or regex) into one element and let the quantifiers ?, +, * operate on the specified group.

Example: (wo)?man -> man, woman

Brackets also come into play when | is used within words.

Examples: m(a|e)n -> man,men
h(i|a|u)t -> hit, hat, hut
(some|any)body -> somebody, anybody
(some|any)(body|one|thing) -> somebody, anybody, someone, anyone, something, anything

3.2.4. Character class

[…] Matches each character listed between the square brackets.
[abc] thus is equivalent to (a|b|c).

Example: wom[ae]n -> woman, women

Note that these principles also apply for mark-up queries in VOICE 3.0 Online.

Example: _[123] -> _1, _2, _3
(i.e. all pauses with a length of one or two or three seconds)

3.2.5. Final remarks on regex

The previous sections introduced the different *building blocks* of regular expression which you can use to formulate fine grained searches in VOICE 3.0 Online.

Though only searches for word forms were provided in our regex-examples so far, regex are by no means restricted to word-form searches. They are also applicable when searching for lemma or POS or even mark-up.

You may have noticed that there are often different ways to achieve the same search results. For instance, any of the following queries will be searching for the words *later*, *latter* and *letter*: (later|latter|letter)
l[ae]tt?er
l(a|e)t{1,2}er

This means that you can choose according to your personal preferences. As a general principle, it is probably a good idea to keep queries as simple as possible.

Talking of simplicity: Don't get overwhelmed by the amount of possibilities you are offered by the regex syntax – and consequently also in VOICE 3.0 Online. In most cases, you will be just using .* or .+ in order to denote "a sequence of random characters" and maybe (xxx|yyy) to denote "alternative sequences".

Part II Examples

4. TOKEN SEARCH (word forms)

This section provides examples for token search (i.e., searching for words and word forms) in VOICE 3.0 Online.

4.1. Simple token search

SEARCH	EXPLANATION	Example Search	Finds
token	Search for a particular **token** **Contracted forms** (e.g., *wanna*, *gonna*, *don't*, *it's*) need to be searched for with a space inserted before the contracted part.	*manage*	*manage*
		wan na	*wanna*
		do n't	*don't*

4.2. Token search with wildcards

SEARCH	EXPLANATION	Example Search	Finds
token.* ***(no space)***	Token plus **zero or more** characters	*.*ment*	*department* *environment* *segment*
		*manage.**	*manage* *MANAGED* *manager* *management*
token.? ***(no space)***	Token plus **zero or one** character	*behave.?*	*behave* *behaves*
token.+ ***(no space)***	Token plus **one or more** characters	*house.+*	*houses* *household*
token[...] ***(no space)***	Token with character classes	*wom[ae]n*	*woman* *women*
token(...)? ***(no space)***	Token with optional components	*open(ness)?*	*open* *openness*

5. POS AND LEMMA SEARCH

This section provides examples for part-of-speech (POS) and lemma search in VOICE 3.0 Online.

5.1. Simple POS search

SEARCH	EXPLANATION	Example Search	Finds
POS **(equivalent to pf:POS)**	All **tokens** with a particular **part-of-speech** tag (POS) in form or function position NB. Simple searches for highly frequent POS tags, e.g. interjections, are likely to yield many hits and might slow down the search engine.	*JJ*	*professional_**JJ**(JJ)* *p_non-formal_PVC(**JJ**)* *full-time_**JJ**(RB)* *present_**JJ**(JJ)/VV(VV)*
p:POS	All **tokens** with a **part-of-speech tag** in **form position**	*p:JJ* (adj. in form position)	*professional_**JJ**(JJ)* *full-time_**JJ**(RB)*
f:POS	All **tokens** with a **part-of-speech tag** in **function position**	*f:JJ* (adj. in function position)	*professional_JJ(**JJ**)* *p_non-formal_PVC(**JJ**)*

5.2. POS search with wildcards

SEARCH	EXPLANATION	Example Search	Finds
POS.*	**POS tag** with **wildcard** NB. Using wildcards with POS tags is meaningful for POS categories that are sub-divided into finer categories, e.g. Verbs, Adjectives, Nouns, Adverbs. Users may also want to narrow down results by adding sub-specifications, e.g. (go,V.* see 8. EXPERT SEARCH).	*V.** Verb-tag with wildcard (all verb forms, e.g. VV, VBP, ...)	*want_VVP(VVP)* *to ask_VV(VV)* *saw_VVD(VVD)* *is_VBZ(VBZ)*
		*J.** Adjective-tag with wildcard (all adjective forms, i.e. JJ, JJR, JJS)	*big_JJ(JJ)* *cheaper_JJR(JJR)* *most_JJS(JJS)*
		*N.** Noun-tag with wildcard (all noun forms, e.g. NN, NNS, ...)	*ideas_NNS(NNS)* *london_NP(NP)* *netherlands_NPS(NPS)*

5.3. Lemma search			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
l:lemma	Finds all **tokens** of a particular **lemma**	*l:walk*	*walk* *walking* *walked*

6. MARK-UP SEARCH - **NEW!!!**

This section provides examples for various mark-up searches in VOICE 3.0 Online.

6.1. Searches for tokenised mark-up			
6.1.1. Pauses			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
_0 **_1** **_2**	Pauses of different **lengths** (Numbers indicate length in seconds as transcribed. _0 indicates a short pause of up to approximately half a second, see VOICE Mark-up conventions.) NB. Be mindful that especially short pauses are rather frequent and are thus best searched in combination with another element (e.g. tokens or POS tags, see 4.4. and below). NB. In order to find pauses irrespective of their length, we recommend you use the former POS tag **PA**, see "Other mark-up searches" and POS short tag set.)	*_0* *_1* *_2*	*(.)* *(1)* *(2)*

6.1.2. Laughter

SEARCH	EXPLANATION	Example Search	Finds
_@ **_@@** **_@@@**	**Laughter**, each @-symbol refers to the respective number of syllables laughed (e.g. Ha ha = @@, see VOICE Mark-up Conventions).	_@@	@@ @@ @@
_@+	**Laughter strings** with at least one "@", i.e. laughter strings with any number of syllables.	_@+	@ @@ @@@@ @@@@@@@
_@{2,4}	**Laughter** with a **defined string length**	*_@{2,4}* *(sequences of minimum 2 and maximum 4 repetitions of the @-character)*	@@ @@@ @@@@

6.2. Searches for mark-up in pointed brackets

6.2.1. Speaking modes

SEARCH	EXPLANATION	Example Search	Finds
<@/> **<fast/>** **<slow/>** **<loud/>** **<soft/>** **…**	***Stretch*** of speech marked <xyz> token token </xyz> NB. For the full list of speaking modes see the VOICE Mark-up Conventions. NB. As an equivalent alternative to <@/> you can also search for <laughingly/>.	*<@/>* *(speaking mode: laughingly)*	*<@> yeah yeah </@>*
		<soft/> *(speaking mode: soft)*	*<soft> okay </soft>*

6.2.2. Non-English Speech

SEARCH	EXPLANATION	Example Search	Finds
<L/>	All stretches of transcription in languages marked as **non-English speech** (L1, LN or LQ; see VOICE Mark-up Conventions).	*<L/>*	*<LNger>* ***diesen leberknoedel*** *{this liver dumpling} </LNger>* *<L1slo>* ***xxxx*** *</L1slo>* *<LQslo>* ***dobre****? {good} </LQslo>*
<L1/>	All stretches of speakers' **first languages** (L1) other than English	*<L1/>*	*<L1mlt>* ***mara*** *{woman} </L1mlt>* *<L1rum>* ***securitate*** *</L1rum>*
<LN/>	All stretches of speech in **neither English nor** a speaker's **first language**	*<LN/>*	*<LNger>* ***senf****. {mustard} </LNger>* *<LNita>* ***toscana****? {name of pizza} </LNita>*
<LQ/>	All stretches of speech where it is **not known** whether they were a speaker's **first** or a **foreign language**	*<LQ/>*	*<LQfre>* ***melange*** *{mixture} </LQfre>* *<LQger>* ***danke*** *{thanks} </LQger>*
<L translation="token"/>	Finds **tokens** in **any translation tag** (L1, LN or LQ)	*<L translation="yes"/>*	*<L1scc>* ***jeste*** *{yes} </L1scc>* *<LNita>* ***s:i.*** *{yes} </LNita>*
<L1 translation="token"/> **<LN translation="token"/>** **<LQ translation="token"/>**	Finds **tokens** in **translations** either an **L1**, **LN** or an **LQ**-tag	*<LN translation="yes"/>*	*<LNger>* ***ja*** *{yes} </LNger>* *<LNita>* ***s:i.*** *{yes} </LNita>* *<LNfre>* ***oui*** *{yes} </LNfre>*
<L1language/> **<LNlanguage/>** **<LQlanguage/>**	Finds and highlights a **stretch** of a **particular language tag** NB: Languages are abbreviated according to the iso 639-2 codes.	*<L1ger/>*	*<L1ger>* ***nein danke*** *{no thanks} </L1ger>*
		<LNita/>	*<Lnita>* ***grazie*** *{thanks} </Lnita>*

6.2.3. Overlaps

SEARCH	EXPLANATION	Example Search	Finds
** **	**Overlaps** NB: This search is best narrowed down, e.g. by using *within* or *containing* (see example on the right and e.g. 8.4.1. Tokens within Mark-up)	* *	*<1>* ***what is it*** *</1>* *<3>* ***yeah*** *</3>* *<6>* ***we have that*** *</6>*
		*okay within *	*<2>* ***okay*** *</2>* *<8> oh* ***okay*** *</8>*

6.2.4. Onomatopoeia

SEARCH	EXPLANATION	Example Search	Finds
<ono/>	**Onomatopoeia**	*<ono/>*	*<ono>* ***wəʊəʊ****: </ono>* *<ono>* ***brbrm*** *</ono>*

6.2.5. Speaker noises

SEARCH	EXPLANATION	Example Search	Finds
<clears throat/> **<whistles/>** **…**	**Speaker noises** NB: For the full list of speaker noises see the VOICE Mark-up Conventions.	*<clears throat/>*	*<clears throat>*

6.2.6. Utterances

SEARCH	EXPLANATION	Example Search	Finds
<u>	**Finds the beginning of an utterance,** useful in combination with other searches, see also 8.3. Using placeholders in phrase search.	*<u> well* (finds *well* at the beginning of an utterance)	***well*** @@
</u>	**Finds the end of an utterance,** useful in combination with other searches, see	*UH </u>* (finds am interjection *UH* at the end of an utterance)	***er***: =

	also 8.3. Using placeholders in phrase search.		
6.3. Searches for mark-up via POS tags or token prefix			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
FW **f_.***	All **'foreign' (i.e. non-English) tokens**	*FW* *f_.**	*<LNbul>* ***rakia_FW(FW)*** *{raki} </LNbul>* *<L1ger>* ***tschuldigung_FW(FW)*** *{sorry} </L1ger>* *<LNger>* ***schottentor?_FW(FW)*** *{place in vienna} </LNger>*
PA	All **pauses** (POS tag PA) NB. The POS tag for pauses (PA) exists in VOICE 2.0 XML POS and continues to be searchable in VOICE 3.0 Online. Due to added tokenization and merged TEI-XML representation, VOICE 3.0 XML represents pauses without a POS tag. NB. Pauses can also be searched for as tokenized mark-up (see 6.1.1. Pauses).	*PA*	*(.)* *(1)* *(4)*
PVC **p_.***	All **pronunciation variations and coinages**	*PVC* *p_.**	*<pvc>* ***creativitly_PVC(NN)/PVC(RB)*** *{creatively} </pvc>* *<pvc>* ***frauding_PVC(VVG)*** *</pvc>*
ONO **o_.***	All onomatopoeia	*ONO* *o_.**	*<ono>* ***bvuff_ONO(ONO)*** *<ono/>* *<ono>* ***lalala_ONO(ONO)*** *</ono>*
SP **s_.***	All **spelt** items NB. While spelt tokens are annotated with different POS tags (e.g. SP, CD, NN), they can be retrieved through the common prefix s_.	*SP* *s_.**	*<spel>* ***p h d_NN(NN)*** *</spel>* *<spel>* ***a_LS(LS)*** *</spel>* *<spel>* ***e u_NP(NP)*** *</spel>* *<spel>* ***a m_RB(RB)*** *</spel>* *<spel>* ***s_p_SP(SP)*** *</spel>*

7. PHRASE SEARCH – partly **NEW**!!!

This section provides examples for different types of phrase searches in VOICE 3.0 Online.

7.1. Lexical phrases (tokens)

SEARCH	EXPLANATION	Example Search	Finds
token token	Finds a particular sequence of lexical tokens / word forms.	*and the*	*and the* *a:nd the* *(and) (1) the* *and hh the* *and the </@> hh and the*

7.2. Part-of-speech and lemma combinations

SEARCH	EXPLANATION	Example Search	Finds
POS1 POS2 POS3	Finds a particular sequence of POS tags	*DT JJ NN* *(*Determiner followed by adjective followed by noun)	*a hu:ge university* *the other way* *a good soccer*
POS1 POS2 lemma1	Finds a particular sequence of POS tags and/or lemma tags	*DT JJ l:university*	*a hu:ge university* *a (.) modern university* *the private universities*

7.3. Token, POS, lemma combinations

SEARCH	EXPLANATION	Example Search	Finds
token POS **POS token** **lemma POS** **token1 POS1 POS2** **POS1 token1 token2** **…**	Finds **sequences** of **tokens**, **POS tags and lemmas**	*whenever PP* *(*token *whenever* plus personal pronoun)	*whenever you* *whenever they* *whenever we*
		PVC er *(*pronunciation variation and coinage plus token *er)*	*<pvc>* ***preferently*** *</pvc> er* *<pvc> (****knowledges****) </pvc> er*

		you MD VV *(*token *you* followed by modal verb and base verb)	*you will go* *you can get*
		play the NN (tokens *play* and *the* followed by singular noun)	*play the card* *play the doorman* *play the map*

7.4. Token and mark-up sequences			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
token <speaking mode/>	**Token** followed by **speaking mode** soft	*yeah <soft/>* (token *yeah* followed by mark-up indicating softly spoken)	**yeah** <soft> **okay okay** <1> **i understand** </1> </soft>
token <L/>	Token followed by non-English speech	*say <L/>* (token *say* followed by any language tag)	*can **say** <LNger> **vermissen** {to miss} </LNger> (.)* *how do you **say** <LNfre> **subvention** {subvention, subsidy} </LNfre>* *now we **say** (.) <L1nor> **trettito** {thirty-two} </L1nor>*
		is <L1/>	***is** <L1ger> **garnisongasse** {street name} </L1ger>*
_@ token	Laughter followed by token/word	*_@+ yes* (any number of laughter-syllables followed by token *yes)*	***@ yes*** ***@@ yes*** ***@@@** <1> **yes** </1>*
token _1	Token followed by pause	*i _1* (token *i* followed by a 1 -second pause)	*no **i (1)** i just* *what **i: (1)** would like to*

7.5. POS and mark-up sequences			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
<@/> POS	Speaking mode followed by POS	*<@/> UH* (laughingly spoken followed by interjection)	*<@> **no** </@> @ **ah*** *<@> **okay** </@> (1) **erm*** *<@> **well** </@> (1) **wow.***
<L/> POS	Tag indicating non-English speech followed by POS	*<L/> PVC* (language tag followed by PVC)	*<L1scc> **xx x** </L1scc> <pvc> **sympatic** </pvc>*
**POS **	POS tag followed by overlap	*UH * (interjection followed by overlap)	***er** <4> **reaction** </4>* ***a:h** <2> **well yes** </2>* ***huh?** (.) <3> **and the:** </3>*

8. EXPERT SEARCH – NEW!!!

This section provides examples for fine-tuning searches in VOICE 3.0 Online.

8.1. Fine-tuning searches (and)			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
,	Meaning: Boolean **AND** Finds **sub-specifications** of tokens with POS tags or lemmas. (Any sequence of items before and after **,** is possible.)		
token,POS	**Token tagged** with a particular **POS tag**	*walk,NN* (token walk as noun)	*a five minute **walk_NN(NN)***
		RB,real (token real as adverb)	***real_RB(RB)** beautiful*
l:lemma,POS	**All tokens** of a particular **lemma tagged with** a particular **POS tag**	*l:go,VVZ* (all tokens with lemma go and tagged with verb-tag present tense 3rd person singular)	*everybody **goes_VVZ(VVZ)*** *who <@> loses **go_V(VVZ)** <8> for drinks*

8.2. Fine-tuning searches (or)			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
|	Meaning: alternation (Boolean **OR**) Finds any of the options to the left or the right of the vertical line **|** . (Any sequence of tokens, lemmas or POS tags before and after | are possible. More than two options can be specified. You can use condition1|condition2|…|conditionN. Round brackets can be used for increasing readability of the query (see Section 3).		
token|token **token|token|token|…**	Finds **either** one of these **tokens**	*say|mean that*	*mean that* *say that*
POS1|POS2	Finds **either** one of these **POS tags**	*VHD|VBD* (verb have or be, past tense)	*was_VBD* *were_VBD* *had_VHD*
token|POS	Finds either this **token** or **POS tag**	*EX|you* (existential *there* or *you*)	*there_EX(EX)* *you_PP(PP)*
l:lemma1|l:lemma2	Finds **either** one of these **lemmas**	l:say|l:mean that (lemma *say* or lemma *mean* plus token that)	*say that* *said that* *saying that* *mean that* *means that*
token|l:lemma	Finds **either** this **token** or this **lemma**	man|l:house	*man* *house* *houses*
token1|token2 l:lemma1	**Token1 or token2** followed by **lemma1**	never|always l:say (token *never* or *always* followed by lemma say) equivalent to: (never|always) l:say	*always say* *never said* *always saying*
token1 POS1|POS2	**Token1** followed by **POS1 or POS2**	i RE|UH	*yeah i*

		(token *i* followed by response marker or interjection)	*er i* *mhm: i*
		always VBZ|VHZ|VVZ (token *always* followed by third pers. singular form of be, have or other verbs)	*always is* *always has* *always depends* *always does*
_@ POS1|POS2	**Laughter** followed by **POS1 or POS2**	_@ UH|RE (one syllable of laughter followed by interjection or response marker)	*@ er* *@ yeah* *@ ah*

8.3. Using placeholders in phrase search			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
.* **.+**	**Placeholder for *any* word.** NB. Technically this is just a normal regex, searching token with at least 0 characters (.*) or at least 1 character. Because there are no zero-character token, these two variants yield exactly the same result: they just match *any* token.	*i .* go* *i .+ go*	*i will go* *i can go* *i must go*
.* .* .* **.+ .+ .+**	Searches for a **specific number of random words**	*i .* .* go*	*i wanted to go* *i decided to go*
.*{number} **{number}**	A more concise way of searching for a specific number of random words. NB. .*{number}, .+{number} and {number} are equivalent. Use the one that fits your purposes best.	*i .*{2} go* *i {2} go*	*i wanted to go* *i decided to go*
.*{minimal,maximal} **{minimal,maximal}**	Specifies a range of minimal and maximal number of random words.	*go .*{1,2} university*	*go to university* *go to the university* *go to state university*
token1 {0,3} token2	**Zero to three** tokens between token1 and token2 in the same utterance	*i {0,3} go*	*i must go* *i decided to go* *i only want to go*
<u> {0,2} token	**A token** which is **situated between zero and two** tokens **after the beginning** of an **utterance.**	*<u> {0,2} well*	EDint331:28 *S1: e:r well* EDsed251:585 *S18: and erm (1) well*

8.4. Search *within*: Find tokens and POS/lemmas within mark-up			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
8.4.1. Tokens within Mark-up			
token *within* <speaking mode/>	**Token** *within* pointed brackets, e.g. **Speaking mode**	*go within <soft/>*	*<soft> have to go: </soft>*
		yeah within <@/>	*<@> yeah yeah yeah </@>*
token *within* <L1/LN/LQ/>	**Token** *within* **tag for non-English speech**	nein *within* <L1ger/>	*<L1ger> nei:n {no:} </L1ger>*
***token within ***	**Token** *within* **overlapping speech**	*really within *	*<3> really strong. (1) hm? </3>* *<4> really? </4>* *<2> not really </2>*
_@ within <speaking mode/>	**Laughter** *within* **speaking mode**	*_@ within <loud/>*	*loud> @ </loud>*
8.4.2. POS within Mark-up			
POS *within* <speaking mode/>	**POS tag** *within* **speaking mode**	*RE within <loud/>*	*<loud> yeah_RE(RE) </loud>* *<loud> okay?_RE(RE) </loud>*
**POS *within* **	**POS tag** *within* **overlap**	*FI within *	*<4> sorry_FI(FI) </4>* *<7> oh_FI(FI) my_FI(FI) gosh_FI(FI) </7>* *<8> you're_FI(FI) welcome_FI(FI) </8>* *<4> bye-bye_FI(FI) </4>*
8.4.3. Lemma within Mark-up			
l:lemma *within* <speaking mode/>	**Lemma** *within* **speaking mode**	*l:be within <imitating/>*	*<imitating> be: the members of the working groups <8>*
**l:lemma *within* **	**Lemma** *within* **overlap tag**	*l:say within *	*<6> say </6>* *<2> am i saying</2>* *<4> he'd say </4>*

8.5. Search *containing*: Find stretches of speech with particular mark-up that contain particular tokens/POS/lemmas			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
8.5.1. Mark-up containing token			
** *containing token***	**Overlap** containing **token**	* containing funny*	*<7> so funny </7>* *<7> a little bit funny </7>* *<6> that's funny</6>*
<L/> *containing* token	**Language tags marked as non-English speech** containing **token**	*<L1/> containing ja*	*<L1ger> ja tust du (weiter) {do you hurry up} </L1ger>* *<L1ger> ja? {yeah} </L1ger>*
<soft/> *containing* token	**Speaking mode** containing **token**	*<soft/> containing okay*	*<soft> okay </soft>* *<soft> okay it's my turn? </soft>*
8.5.2. Mark-up containing POS			
<speaking mode/> *containing* POS	**Speaking mode** containing **POS**	*<loud/> containing RE*	*<loud> no don't </loud>* *<loud> yeah. </loud>* *<loud> yes </loud>* *<loud> okay there is coffee </loud>*
8.5.3. Mark-up containing lemma			
<@/> *containing* l:lemma	**Speaking mode laughingly spoken** containing **lemma**	*<@/> containing l:go*	*<@> when and where to go </@>* *<@> you went shopping </@>*

8.6. Combined searches with wildcards and fine-tuning			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
token .* .* .*	**Token** plus **placeholders** for any number of unspecified tokens	*i really .* .* .** *i really .*{3}* *i really {3}* *("i really" followed by 3 random token)*	*i really feel so old* *i really appreciate talking to* *i really think that you*
.* token .*	**Token preceded** by **placeholder** and **followed** by **placeholder**	*.* i really .* .** *{1} i really {2}* *(" i really" preceded by 1 and followed by 2 random lexical tokens)*	*what i really liked was* *e:r i really hope that* *i i really don't*
p:POS,f:POS	**Combination** of particular **form** and **function POS tags**	*p:JJ,f:RB* *(token tagged adjective in form-position and adverb in function position)*	*you grew up (.) bilingual_JJ(RB).* *perform good_JJ(RB) in another language*
token.*,POS **.*token,POS**	**Token with wildcard** tagged with a particular **POS tag**	*thank.*,FI* *(*thank with wildcard as formulaic item)	*thanks* *thank you*
		*.*ness,PVC* (all tokens ending in -ness tagged PVC)	*competiveness* *healthness* *europeanness* *business*
l:lemma,POS.*	All instances of a lemma tagged with a particular POS tag with wildcard. NB. In phrases, this type of search can be useful to retrieve all POS tags of a superordinate POS category, e.g. V.* (all verbs), N.* for (all nouns).	*l:see,V.** (run, all verb-forms)	*it runs_VVZ(VVZ) again* *you just kind of k- run_VVP(VVP) through* *we're running_VVG(VVG) out of time*
token1 POS1 token2 .*	**Combinations** of **token** and **POS tag** plus a **wildcard** (standing for any token)	*i RB think .**	*i also_RB think that*

		(*i* followed by adverb followed by *think* followed by any token)	
POS1 POS2 token,POS.*	**Sequence** of **POS tag**s followed by a **token** with a **sub-specification**	*PP RB think,V.** (Personal pronoun followed by adverb followed by *think* as verb)	*i_PP also_RB think_VVP* *could you_PP maybe_RB think_VV*
l:lemma,POS.*	**Token** of a particular **lemma sub-specified** with **POS tag**	*l:show,V.** (Lemma *show* tagged as verb-form)	*show* *showing* *showed* *shown*
		*l:thought,NN.** (Lemma *thought* as singular or plural noun)	*thoughts* *thought*
l:lemma.*,POS.*	**Token** of **a lemma** with **wildcard sub-specified** with **POS tag** with **wildcard**.	*l:re.*,V.** (Lemma starting with *re-* tagged as any verb-form)	*recording* *read* *related* *registering*
POS l:lemma POS,.*token	**POS tag** followed by **lemma** followed by **POS tag sub-specified** with a **token** with **wildcard**	*DT l:good NN,.*ion* (Determiner followed by lemma *good* followed by a noun ending in *-ion*)	*a better situation* *the: good discussion* *the best solution*
POS1|POS2 token1	**Either POS tag1** or **POS tag2** followed by **token1**	*RB|JJ good* (Adverb or adjective followed by *good)*	*very good* *no good* *good good* *many good*
token1|token2|token3 POS1	**Either token1**, **token2** or **token3** followed by **POS tag1**	*yes|yeah|yah UH* (Tokens *yes*, *yeah* or *yah* followed by POS category interjection)	*yes o:h* *yah? er* *yeah. ooph*

8.7. Combined searches with *within* or *containing*			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
token *within* <speaking mode/>	**Token** *within* **speaking mode**	*well within <soft/>* (well within tag indicating softly spoken)	*<7> <soft> well you know </soft> </7>* *<soft> mhm (2) very well </soft>* *<soft> on Thursday as well </soft>*
**POS1|POS2|POS3 *within* **	Either **POS tag1, 2** or **3** *within* **overlap**-tag	*FI|RE|UH within * (Formulaic item or response marker or interjection within overlap tag)	*<3> thanks_FI(FI) </3>* *<5> ye:s_RE(RE) </5>* *<10> er:_UH(UH) </10>*
**laughter *within* **	**Laughter** *within* **overlap**-tag	*_@@ within * (Two syllables of laughter within overlap-tag)	*<8> @@ </8>* *<1> hi @@ </1>*
<speaking mode/> *containing* token,POS	**Speaking mode** tag *containing* a **token sub-specified** with a POS tag	*<soft/> containing well,DM* (Speaking mode soft containing token *well* POS tagged as discourse marker)	*<soft> well_DM(DM) you know </soft>* *<soft> well_DM(DM) (then) yeah of course but </soft>*
** *containing* token,POS**	**Overlap**-tag *containing* a **token sub-specified** with a **POS tag**	* containing you,FI* (Overlap containing token *you* tagged as formulaic item)	*<6> thank you_FI(FI) @@@ </6>* *<11> see you_FI(FI) </11>* *<7> you_FI(FI)'re welcome </7>*
** *containing* l:lemma,POS.***	**Overlap**-tag *containing* a **token** of a **lemma sub-specified** with a **POS tag** with **wildcard**	* containing l:good,RB.** (Overlap tag containing a token of a lemma *good* as any type of adverb, i.e. RB,RBR,RBS)	*is going <6> (good)_RB(RB) </6>* *<1> much better_RBR(RBR) </1>*

8.8. Combined searches with placeholder			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
token1 {0,1} POS1 POS2	**Token1** followed by a defined **range** of **context** followed by **POS tag1** and **2**	*the {0,2} JJ NN* (Token *the* followed by zero to two random tokens followed by adjective and noun)	*the main building* *the second third lesson* *the the legal stuff* *the legal erm legal clinic*
<speaking mode/> *containing* token {0,1}	**Speaking mode** tag containing a **token** followed by a defined **range** of tokens	*<soft/> containing yes {1,5} UH* (<Speaking mode/> containing token *yes* followed by a range of 1 to 5 random tokens and POS tag UH.)	*<soft> yes okay </soft>* *<soft> a:h yes. (.) [name2]</soft>* *<soft> yes they must be calibrated </soft>*

8.9. Combined mark-up searches			
SEARCH	**EXPLANATION**	**Example Search**	**Finds**
PA <speaking mode/>	Any **pause** followed by a **speaking** mode tag	*PA <fast/>* (Pause of any length followed by fast speech)	*(.) <fast> keep that in mind </fast>*
PVC within <speaking mode/>	All **pronunciation variations and coinages** which occur **within a speaking mode tag**	*PVC within <soft/>*	*<soft> <pvc> unconcrete </pvc> </soft>* *<soft> a balloon <pvc> wobbler? </pvc> </soft>*
** containing SP**	All **overlaps** containing **spelt** tokens	* containing SP*	*<9> <spel> s p </spel> </9>*

9. Links

VOICE CLARIAH project & new VOICE website: https://voice.acdh.oeaw.ac.at/

Access to VOICE 3.0 Online: https://voice3.acdh.oeaw.ac.at/

How to cite VOICE 3.0 Online:

Recommended full citation for VOICE 3.0 Online:
VOICE. 2021. *The Vienna-Oxford International Corpus of English* (version VOICE 3.0 Online). Founding director: Barbara Seidlhofer; Principal investigators VOICE 3.0: Marie-Luise Pitzl, Daniel Schopper; Researchers: Angelika Breiteneder, Hans-Christian Breuer, Nora Dorn, Theresa Klimpfinger, Stefan Majewski, Ruth Osimk-Teasdale, Hannes Pirker, Marie-Luise Pitzl, Michael Radeka, Stefanie Riegler, Barbara Seidlhofer, Omar Siam, Daniel Stoxreiter. https://voice3.acdh.oeaw.ac.at (date of last access).

Short citation for VOICE 3.0 Online:
VOICE. 2021. *The Vienna-Oxford International Corpus of English* (version VOICE 3.0 Online). https://voice3.acdh.oeaw.ac.at (date of last access).

Search Manual for VOICE 3.0 Online:
https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/09/Search-manual-VOICE-3.0-Online.pdf

How to cite the *Search manual for VOICE 3.0 Online*:

Osimk-Teasdale, Ruth; Pirker, Hannes; Pitzl, Marie-Luise. 2021. *Search manual for VOICE 3.0 Online.* https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/09/Search-manual-VOICE-3.0-Online.pdf (date of last access).

VOICE Mark-up conventions:
https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/04/VOICE-mark-up-conventions.pdf

Recommended citation:
VOICE Project. 2007. "Mark-up conventions". *VOICE Transcription Conventions [2.1]*. https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/04/VOICE-mark-up-conventions.pdf (date of last access).

VOICE Spelling conventions:
https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/04/VOICE-spelling-conventions.pdf

Recommended citation:
VOICE Project. 2007. "Spelling conventions". *VOICE Transcription Conventions [2.1]*. https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/04/VOICE-spelling-conventions.pdf (date of last access).

VOICE Part-of-Speech Tagging Manual:
https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/04/POS-tagging-and-lemmatization-manual.pdf

Recommended citation:
VOICE Project. 2014. *VOICE Part-of-Speech Tagging and Lemmatization Manual*.

https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/04/POS-tagging-and-lemmatization-manual.pdf (date of last access).

VOICE Short POS Tagset:
https://voice.acdh.oeaw.ac.at/wp-content/uploads/2021/04/Short-POS-tagset.pdf