Kostenloses eBook

# LERNEN

# grep

Free unaffiliated eBook created from
**Stack Overflow contributors.**

#grep

# Inhaltsverzeichnis

# Über

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: grep

It is an unofficial and free grep ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official grep.



Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

# Kapitel 1: Erste Schritte mit grep

## Bemerkungen

grep druckt Zeilen, die eine Übereinstimmung für ein Muster in Dateien enthalten.

grep kann reguläre Ausdrücke verwenden und bietet mehrere Optionen zur Verbesserung der Ergebnisqualität.

## Verweise

- POSIX
- FreeBSD-Manpage
- OpenBSD-Manpage
- GNU grep Online-Handbuch
- Illumos-Manpage

## Versionen

# POSIX grep

| Ausführung | Veröffentlichungsdatum |
|---|---|
| POSIX.2 | 1992-01-01 |
| IEEE Std 1003.1-2001 | 2001-12-06 |
| IEEE Std 1003.1, Ausgabe 2004 | 2004-01-01 |
| IEEE Std 1003.1, Ausgabe 2013 | 2013-04-19 |
| IEEE Std 1003.1, 2016 Ausgabe | 2016-09-30 |

# Illumos / OpenSolaris grep

| Ausführung | Veröffentlichungsdatum |
|---|---|
| 2005-06-14 | 2005-06-14 |

| Ausführung | Veröffentlichungsdatum |
|---|---|
| 2005-09-06 | 2005-09-06 |
| 2012-03-30 | 2012-03-30 |
| 2012-09-17 | 2012-09-17 |
| 2013-05-14 | 2013-05-14 |

# GNU grep

| Ausführung | Veröffentlichungsdatum |
|---|---|
| 2,0 | 1996-10-01 |
| 2.2 | 1998-04-27 |
| 2.3 | 1999-02-14 |
| 2.4.1 | 2000-03-01 |
| 2.4.2 | 2000-03-09 |
| 2.4 | 1999-12-03 |
| 2.5.1 | 2004-10-29 |
| 2.5.1a | 2004-11-19 |
| 2.5.3 | 2007-08-02 |
| 2.5.4 | 2009-02-09 |
| 2,5 | 2002-03-13 |
| 2.6.1 | 2010-03-25 |
| 2.6.2 | 2010-03-29 |
| 2.6.3 | 2010-04-02 |
| 2.6 | 2010-03-23 |
| 2,7 | 2010-09-20 |
| 2.8 | 2011-05-13 |
| 2,9 | 2011-06-21 |

| Ausführung | Veröffentlichungsdatum |
|---|---|
| 2.10 | 2011-11-16 |
| 2.11 | 2012-03-02 |
| 2.12 | 2012-04-23 |
| 2.13 | 2012-07-04 |
| 2.14 | 2012-08-20 |
| 2.15 | 2013-10-26 |
| 2.16 | 2014-01-01 |
| 2.17 | 2014-02-17 |
| 2.18 | 2014-02-20 |
| 2.19 | 2014-05-22 |
| 2,20 | 2014-06-03 |
| 2.21 | 2014-11-23 |
| 2,22 | 2015-11-01 |
| 2.23 | 2016-02-04 |
| 2.24 | 2016-03-10 |
| 2,25 | 2016-04-21 |
| 2.26 | 2016-10-02 |
| 2.27 | 2016-12-06 |
| 2,28 | 2017-02-06 |

## BSD grep / FreeGrep

| Ausführung | Veröffentlichungsdatum |
|---|---|
| OpenBSD 3.0 | 2001-12-01 |
| OpenBSD 3.4 | 2003-11-01 |
| OpenBSD 3.5 | 2004-05-01 |

| Ausführung | Veröffentlichungsdatum |
|---|---|
| OpenBSD 3.6 | 2004-11-01 |
| OpenBSD 3.7 | 2005-05-19 |
| OpenBSD 3.8 | 2005-11-01 |
| OpenBSD 3.9 | 2006-05-01 |
| OpenBSD 4.0 | 2006-11-01 |
| OpenBSD 4.1 | 2007-05-01 |
| OpenBSD 4.3 | 2008-05-01 |
| OpenBSD 4.8 | 2010-11-01 |
| OpenBSD 5.0 | 2011-11-01 |
| OpenBSD 5.3 | 2013-05-01 |
| OpenBSD 5.7 | 2015-05-01 |
| OpenBSD 5.8 | 2015-10-18 |
| OpenBSD 5.9 | 2016-03-29 |
| NetBSD 2.0 | 2004-12-09 |
| NetBSD 4.0 | 2007-12-19 |
| NetBSD 6.0 | 2012-10-17 |
| NetBSD 7.0 | 2015-09-25 |
| FreeBSD 9.0 | 2012-01-02 |
| FreeBSD 10.0 | 2014-01-16 |

# Examples

## Grundlegende Verwendung

Den Befehl ausführen:

```
grep sam someFile.txt
```

Wenn `someFile.txt` enthält:

```
fred 14 m foo
sam 68 m bar
christina 83 f baz
bob 22 m qux
Sam 41 m quux
```

Erzeugt diese Ausgabe:

```
sam 68 m bar
```

## Fall ignorieren

Gegeben ein `sample` :

```
hello
Hello
HELLO_there
```

Ein normaler `grep` für "Hallo" gibt zurück:

```
$ grep "hello" sample
hello
```

Mit `-i` können Sie Groß- und Kleinschreibung ignorieren und jedes "Hallo" abgleichen:

```
$ grep -i "hello" sample
hello
Hello
HELLO_there
```

## Stimmen Sie ganze Wörter ab

Gegeben ein `sample` :

```
hello world
ahello here
hello_there
```

Ein normaler `grep` für "Hallo" gibt zurück:

```
$ grep hello sample
hello world
ahello here
hello_there
```

Mit `-w` können Sie die Zeilen auswählen, die Übereinstimmungen enthalten, die ganze Wörter bilden:

```
$ grep -w hello sample
hello world
```

### Suchen Sie Text innerhalb eines bestimmten Verzeichnisses rekursiv

## Verwenden von GNU grep

```
grep -r 'pattern' <directory path>
```

Um auch Zeilennummern von Übereinstimmungen `-n` Option `-n`

```
grep -rn 'pattern' <directory path>
```

Nur Dateien mit bestimmten Glob- Mustern suchen

```
grep --include='*.txt' -r 'pattern' <directory path>
```

Dateimuster oder Verzeichnisse ausschließen

```
grep -R --exclude=*.log 'pattern' <directory path>
grep -R --exclude={*.log,*.class} 'pattern' <directory path>

grep -R --exclude-dir=tmp 'pattern' <directory path>
grep -R --exclude-dir={tmp,lib} 'pattern' <directory path>
```

**Hinweise und andere nützliche Optionen**

- `<directory path>` kann übersprungen werden, wenn im aktuellen Verzeichnis gesucht wird
- Die Option `-R` folgt allen symbolischen Links, im Gegensatz zu `-r` das symbolischen Links nur dann folgt, wenn sie sich in der Befehlszeile befinden
- `-l` um nur übereinstimmende Dateien aufzulisten
- `-h` um das Dateinamen-Präfix zu unterdrücken
- `--color=auto` , um übereinstimmende Muster hervorzuheben
- `-m <num>` , um die maximale Anzahl von Übereinstimmungen für jede Dateieingabe anzugeben

## POSIX-Problemumgehung für die rekursive Suche

```
find <directory path> -type f -exec grep -l 'pattern' {} +
```

- Optionen wie `-n` , `-l` usw. können nach Bedarf verwendet werden
- Wenn `{} +` nicht unterstützt wird, verwenden Sie `{} \;` stattdessen
- Weitere Informationen zum `find` finden Sie in der Dokumentation zur `find` wie das Einschließen / Ausschließen von Dateitypen, Verzeichnissen usw.

### Druckt nur den übereinstimmenden Teil der Zeilen

```
echo "Prints only the matching part of the lines" | grep -o "matching"
# prints matching
```

## Grep-Kontextsteuerung

Gegeben ein Datei-Beispiel namens movieslist.

```
Troy
Gladiator
Robin Hood
King Arthur
BraveHeart
The Last Samurai
```

Normaler Grep kehrt zurück

```
grep "Gladiator" movieslist
Gladiator
```

Verwenden Sie nun grep, um die Zeilen über oder unter der Datei zu drucken.

### So drucken Sie die untere Zeile

```
grep -A 1 Gladiator movieslist
Gladiator
Robin Hood
```

### So drucken Sie die obige Zeile

```
grep -B 1 Gladiator movieslist
Troy
Gladiator
```

### Beides drucken

```
grep -C 1 Gladiator movieslist
Troy
Gladiator
Robin Hood
```

Erste Schritte mit grep online lesen: https://riptutorial.com/de/grep/topic/2198/erste-schritte-mit-grep

# Kapitel 2: Kontextzeilensteuerung

## Bemerkungen

`-A` , `-B` und `-C` stehen in POSIX nicht zur Verfügung (siehe POSIX-Spezifikationen für `grep` ).

## Examples

### Zeilen vor und / oder nach übereinstimmendem Muster drucken

Normalerweise druckt `grep` nur passende Zeilen. Im folgenden Beispiel generiert `seq 9` eine Liste mit Zahlen von 1 bis 9, eine pro Zeile, und `grep` druckt eine einzige übereinstimmende Zeile:

```
seq 9 | grep 5
# 5
```

Die Option `-C n` (oder `--context=n` in langer Form) druckt `n` Zeilen vor und nach jeder übereinstimmenden Zeile zusätzlich zu der entsprechenden Zeile selbst:

```
seq 9 | grep -C 2 '5'
# 3
# 4
# 5
# 6
# 7
```

Natürlich werden weniger als `n` Zeilen gedruckt, wenn das Dateiende oder der Dateianfang erreicht wird.

Wenn wir wollen , Linien drucken , nur vor oder erst nach, aber nicht beide, können wir verwenden `-B n` ( `--before-context=n` ) oder `-A n` ( `--after-context=n` ):

```
seq 9 | grep -B 2 '5'
# 3
# 4
# 5

seq 9 | grep -A 2 '5'
# 5
# 6
# 7
```

Beachten Sie, dass diese Optionen in POSIX nicht verfügbar sind (siehe POSIX-Spezifikationen für `grep` ).

Wenn sich die Kontexte von zwei oder mehr übereinstimmenden Zeilen überlappen, werden alle Zeilen zusammen als ein großer Kontext gedruckt. Im folgenden Beispiel ist `5` Teil des Kontexts von `3` und `7` :

```
seq 9 | grep -E --context=2 '3|7'
# 1
# 2
# 3
# 4
# 5
# 6
# 7
# 8
# 9
```

Wenn sich die Kontexte jedoch nicht überlappen, werden sie mit einer Gruppentrennlinie gedruckt. Standardmäßig ist dies ein doppelter Bindestrich ( `--` ):

```
seq 9 | grep -E --context=2 '2|8'
# 1
# 2
# 3
# 4
# --
# 6
# 7
# 8
# 9
```

Wir können eine andere Gruppentrennlinie mit der Option `--group-separator=SEP` oder diese Zeile mit der Option `--no-group-separator` vollständig unterdrücken:

```
seq 9 | grep -E --context=0 --group-separator='****' '2|8'
# 2
# ****
# 8

seq 9 | grep -E --context=0 --group-separator='' '2|8'
# 2
#
# 8

seq 9 | grep -E --context=0 --no-group-separator '2|8'
# 2
# 8
```

Wenn wir schließlich die Option `-v` wählen, um nicht übereinstimmende Zeilen zu drucken, wird stattdessen ein Kontext um diese Zeilen bereitgestellt:

```
seq 9 | grep -E -v '1|3|4|5|6|7|9'
# 2
# --
# 8

seq 9 | grep -E -v -C 1 '1|3|4|5|6|7|9'
# 1
# 2
# 3
# --
# 7
```

```
# 8
# 9
```

Kontextzeilensteuerung online lesen:
https://riptutorial.com/de/grep/topic/4152/kontextzeilensteuerung

# Kapitel 3: Reguläre Ausdrücke

## Examples

### Reguläre Ausdrücke

Das Suchmuster kann auch ein regulärer Ausdruck sein. Laufen:

```
grep '^[A-Z]' someFile.txt
```

Wenn `someFile.txt` enthält:

```
fred 14 m foo
sam 68 m bar
christina 83 f baz
bob 22 m qux
Sam 41 m quux
```

Erzeugt die Ausgabe:

```
Sam 41 m quux
```

da dies die einzige Zeile in `someFile.txt` die mit einem `someFile.txt` beginnt.

### Schau hinter dich

Gegeben die folgende Datei:

```
hello how are you
i am fine
let's go, you!
let's go, baby!
```

`grep` with look-behind erlaubt nur das Drucken einiger Teile:

```
$ grep -Po "(?<=let's go, ).*" file
you!
baby!
```

In diesem Fall stimmt es mit dem überein, was nach "Los geht's" passiert.

Reguläre Ausdrücke online lesen: https://riptutorial.com/de/grep/topic/4183/regulare-ausdrucke

# Kapitel 4: Unterschied zwischen grep, egrep, fgrep, pgrep.

## Einführung

**grep** , **egrep** , **fgrep** , **rgrep** , **pgrep** - sind Befehle in Unix-ähnlichen Betriebssystemen, die Zeilen drucken, die einem Muster entsprechen. Der grep sucht die genannten *Eingabedateien* für Linien eine Übereinstimmung mit dem vorgegebenen *Muster* enthalten. Standardmäßig werden die übereinstimmenden Zeilen gedruckt. Außerdem sind die Variantenprogramme **egrep** , **fgrep** und rgrep identisch mit **grep -E** , **grep -F** und **grep -r** . Diese Varianten sind veraltet, werden jedoch aus Gründen der Abwärtskompatibilität bereitgestellt.

## Syntax

- `grep [OPTIONS] PATTERN [FILE...]`
- `grep [OPTIONS] [-e PATTERN]... [-f FILE]... [FILE...]`

## Parameter

| Symbol | Details Grundlegende reguläre Ausdrücke (BRE) |
|---|---|
| `^` | Mit dem Zirkumflex wird der Zeilenanfang abgeglichen. |
| `$` | wird verwendet, um das Ende einer Zeile anzupassen. |
| `.` | stimmt mit jedem Zeichen außer einer neuen Zeile überein. |
| `[]` | entspricht einem einzelnen Zeichen in den Klammern. Wenn ein `^` drin ist, würde es mit den Zeichen in der Klammer übereinstimmen. |
| `\` | bevor eines der nicht-alphanumerischen Zeichen sie zitiert. |
| `*` | Das Symbol stimmt mit dem vorangehenden Zeichen oder Unterausdruck ein- oder mehrmals überein. |
| `\1` | Die Rückverweise 1-9 stimmen mit dem genauen Text der entsprechenden Gruppe überein. |
| `\{m,n\}` | stimmt mit den vorhergehenden Elementen mindestens *m* und nicht mehr als *n* mal überein. |
| `\|` | `foo\|bar` entspricht foo oder bar. |
| `\?` | Abkürzung für {0,1} |

| Symbol | Details Grundlegende reguläre Ausdrücke (BRE) |
|---|---|
| `\+` | (Abkürzung für **{1,}** ) stimmen höchstens 1 Mal mit dem vorhergehenden Zeichen oder Unterausdruck überein. |
| `\n` | entspricht einer Zeilenumbruchzeile, `\t` entspricht einer Registerkarte usw. |
| `\w` | passt zu jedem Wortbestandteil und \ W zu jedem Zeichen, das kein Wortbestandteil ist. |
| `\<\>` | stimmt mit der leeren Zeichenfolge nur am Anfang oder Ende eines Wortes überein |
| `\b` | passt entweder zu und **\ B** passt wo \ b nicht. |
| **Symbol** | **Details Erweiterte reguläre Ausdrücke (ERE)** |
| `^` | Spiel nur am Anfang |
| `$` | Übereinstimmung nur am Ende einer Zeile. |
| `.` | entspricht einem beliebigen Zeichen (oder einem Zeichen außer einem Zeilenumbruch). |
| `[…]` | stimmt mit einem beliebigen Zeichen überein, das in den Klammern (Zeichensatz) aufgeführt ist. Fügen Sie ein erstes ^ hinzu, und die Bereiche funktionieren wie in BRE (siehe oben). |
| `(…)` | Syntaxgruppe, zur Verwendung mit * oder \ DIGIT-Ersetzungen. |
| `\|` | zur Abwechslung: `foo` \| `bar` entspricht foo oder bar. |
| `*` | stimmt mehrmals mit dem vorhergehenden Zeichen oder Unterausdruck überein: 0, 1 oder mehr Male |
| `+` | stimmt mindestens einmal mit dem vorhergehenden Zeichen überein. |
| `?` | entspricht den vorhergehenden Zeichen 0 oder 1 Mal. |
| `\` | Backslash setzt das nächste Zeichen in Anführungszeichen, wenn es nicht alphanumerisch ist. |
| `{m,n}` | stimmt mit dem vorhergehenden Zeichen oder Unterausdruck zwischen m und n mal überein (fehlt bei einigen Implementierungen); n oder m können weggelassen werden und `{m}` bedeutet genau m |

## Bemerkungen

**fgrep** steht für "Fixed-String Global Regular Expressions Print". **fgrep** ist das gleiche wie **`grep -F`** .

Dieser Befehl ist ein schnelleres grep und verhält sich wie grep, erkennt jedoch KEINE regulären Ausdrücke als Sonderzeichen. Die Suche wird schneller abgeschlossen, da nur ein einfacher String und kein komplexes Muster verarbeitet wird.

---

**pgrep** ist eine Abkürzung für "Process-ID Global Regular Expressions Print". pgrep durchsucht die aktuell laufenden Prozesse und listet die Prozess-IDs auf, die den Auswahlkriterien für stdout entsprechen. pgrep ist praktisch, wenn Sie nur die Prozess-ID-Ganzzahl eines Prozesses kennen möchten.

| grep | egrep (grep -E) | fgrep (grep -F) | pgrep |
|---|---|---|---|
| Grundlegende reguläre Ausdrücke (BRE) | Erweiterte reguläre Ausdrücke (ERE) | Sucht nur Zeichenfolgen | Sucht den Prozess nach Namen |

Für weitere Informationen und Verweise verwenden Sie einige der folgenden Links:

Was ist der Unterschied zwischen grep, egrep und fgrep? Unix und Linux StackExchange

Warum funktioniert mein regulärer Ausdruck in X, aber nicht in Y? Unix und Linux StackExchange

Was ist der Unterschied zwischen grep, pgrep, egrep, fgrep? Superuser

# Examples

## grep mit einfachen regulären Ausdrücken

```
$ grep root /etc/passwd
root:x:0:0:root:/root:/bin/bash
operator:x:11:0:operator:/root:/sbin/nologin
```

## egrep mit erweiterten regulären Ausdrücken

```
$ egrep '^(0|1)+ [a-zA-Z]+$' searchfile.txt
011 AaBBS
```

## fgrep ohne reguläre Ausdrücke

```
$ fgrep "." .bashrc
# will match lines with a dot.
```

## pgrep mit Name des Prozesses

```
$ pgrep python
1299
```

Unterschied zwischen grep, egrep, fgrep, pgrep. online lesen:
https://riptutorial.com/de/grep/topic/8936/unterschied-zwischen-grep--egrep--fgrep--pgrep-

# Credits

| S. No | Kapitel | Contributors |
|---|---|---|
| 1 | Erste Schritte mit grep | Batsu, Benjamin W., Community, David Pärsson, depperm, dingalapadum, fedorqui, Jerry Jeremiah, kdhp, mszymborski, Stuxnet78, Sundeep, UNagaswamy |
| 2 | Kontextzeilensteuerung | Benjamin W., fedorqui, ghostarbeiter |
| 3 | Reguläre Ausdrücke | David Pärsson, dingalapadum, fedorqui |
| 4 | Unterschied zwischen grep, egrep, fgrep, pgrep. | Bor |