

# entr

## Table Of Contents

- [1 Anwendung und Nutzen](#)
- [2 Grundlegende Nutzung von entr](#)
- [3 Praktische Hinweise](#)

Der Befehl `entr` (kurz für Event Notify Test Runner) ist ein kleines, praktisches Linux-Tool, das darauf spezialisiert ist, Dateisystemänderungen zu überwachen und automatisch einen Befehl auszuführen, wenn Änderungen erkannt werden. Dies ist besonders nützlich für Entwickler, die z. B. ihren Code bei Änderungen sofort neu bauen oder Tests ausführen möchten, ohne den Befehl manuell immer wieder ausführen zu müssen.

Der Befehl `entr` (kurz für Event Notify Test Runner) ist ein kleines, praktisches Linux-Tool, das darauf spezialisiert ist, Dateisystemänderungen zu überwachen und automatisch einen Befehl auszuführen, wenn Änderungen erkannt werden. Dies ist besonders nützlich für Entwickler, die z. B. ihren Code bei Änderungen sofort neu bauen oder Tests ausführen möchten, ohne den Befehl manuell immer wieder ausführen zu müssen.

## 1 Anwendung und Nutzen

Automatische Skriptausführung: Immer wenn eine Datei, die überwacht wird, verändert wird (z. B. durch Speichern im Editor), führt `entr` den angegebenen Befehl erneut aus. Dies ist nützlich bei Entwicklungsaufgaben, bei denen Code regelmäßig neu kompiliert oder getestet werden muss.

Dateien beobachten: Mit `entr` kann man bestimmte Dateien beobachten und so z. B. automatische Tests starten, wenn eine bestimmte Datei bearbeitet wird.

## 2 Grundlegende Nutzung von entr

Hier ein einfaches Beispiel:

Code

```
ls *.c | entr gcc -o mein_programm main.c
```

### Erklärung:

Der Befehl `ls *.c` listet alle C-Dateien im Verzeichnis auf und gibt sie an `entr` weiter.

`entr` überwacht nun alle gelisteten Dateien. Sobald eine dieser Dateien verändert wird, führt `entr` den folgenden Befehl aus: `gcc -o mein_programm main.c`.

Das bedeutet, dass `mein_programm` jedes Mal neu kompiliert wird, sobald eine der Dateien geändert wird.

### Ein weiteres Beispiel mit einer Datei

Ein häufiges Beispiel für die Verwendung von `entr` ist die Überwachung eines einzelnen Skripts und das sofortige Ausführen nach Änderungen:

Code

```
echo "mein_skript.sh" | entr bash mein_skript.sh
```

### 3 Praktische Hinweise

Verwendung mit `find`: Falls du rekursiv nach bestimmten Dateitypen suchen möchtest, kann `find` kombiniert werden, z. B.:

Code

```
find . -name "*.py" | entr python main.py
```

Dieser Befehl führt `python main.py` aus, wenn sich eine `.py`-Datei im aktuellen Verzeichnis oder Unterverzeichnis ändert.

Verwendung mit einer Datei oder Liste: Man kann auch eine Liste von Dateien überwachen, indem man sie direkt an `entr` übergibt:

Code

```
echo "datei1.txt datei2.txt" | entr -p cat datei1.txt datei2.txt
```

Die Option `-p` bewirkt hier, dass der Befehl immer wieder ausgeführt wird, auch wenn die Änderung mehrfach erfolgt.

Mit `entr` kann also automatisch auf Dateiänderungen reagiert werden – ideal für eine Umgebung, in der du Skripte oder Programme kontinuierlich testen oder ausführen möchtest!