

<a href="http://ovm-kassel.de">http://ovm-kassel.de</a>   Lernjob	
Lernjob IT-BS-VM-LJ-3.1 Dokumentieren mit Script und AsciiDoc	
Code	IT-BS-VM-LJ-3.1
Autor	André Bauer <a(dot)bauer(at)ovm-kassel(dot)de>
Datum	22. Mai 2018
Lizenz	 Dieses Werk ist lizenziert unter einer <a href="#">Creative Commons Namensnennung - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz</a> .

# Dokumentieren mit Script und AsciiDoc

## 1. Shell-Sessions mit Script documentieren

Shell-Sessions können unter Linux mit dem Programm Script aufgezeichnet werden.

*Beispiel 1. Aufzeichnung einer Shell-Session mit Script*

```
$ script session1.log ①
Skript gestartet, die Datei ist session1.log
$ ssh user@192.168.178.28
user@192.168.178.28's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-28-generic x86_64)

Last login: Thu May 10 18:58:28 2018 from 192.168.178.21
user@ubuntu-vm:~$ ls
Bilder      Downloads  Öffentlich  srv          Vorlagen
Dokumente  Musik      Schreibtisch Videos
user@ubuntu-vm:~$ Abgemeldet ②
Connection to 192.168.178.28 closed.
[user@hostsystem ~]$ exit ③
exit
Skript wurde beendet, die Datei ist session1.log
```

① Startet Script und speichert die Aufzeichnung in der Datei `session1.log`.

② Die SSH-Session wird mit `Strg+D` oder `exit` beendet.

③ Script wird mit `exit` beendet.

Das Ergebnis kann mit `cat` angezeigt werden oder mit einem Texteditor wie z. B. GNU nano, gedit,

emacs, vim, atom oder anderen bearbeitet, kommentiert und z. B. Teil eines AsciiDoc-Dokuments werden (s. u.).

### Beispiel 2. Ausgabe der mit Script gespeicherten Shell-Session

```
$ cat session1.log
Script started on 2018-05-11 08:03:35+02:00
[user@hostsystem ~]$ ssh user@192.168.178.28
user@192.168.178.28's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-28-generic x86_64)

Last login: Thu May 10 18:58:28 2018 from 192.168.178.21
user@ubuntu-vm:~$ ls
Bilder      Downloads  Öffentlich  srv         Vorlagen
Dokumente  Musik      Schreibtisch Videos
user@ubuntu-vm:~$ Abgemeldet
Connection to 192.168.178.28 closed.
[user@hostsystem ~]$ exit
exit

Script done on 2018-05-11 08:04:13+02:00
```

## Monochrome Ausgabe der Shell



In vielen Linux-Distributionen enthält die Ausgabe im Terminal farbige Hervorhebungen. Diese führen zur Ausgabe von unerwünschten Zeichen in den Log-Dateien von Script. Die farbige Ausgabe wird in der Konfigurationsdatei der Shell gesteuert. Bei der Bash ist `~/.bashrc` diese Konfigurationsdatei. Um die farbige Ausgabe von Script zu unterbinden müssen mit einem Texteditor wie `nano` zwei Zeilen verändert werden. Änderungen in der `~/.bashrc` sind in allen neu gestarteten Terminals wirksam. In dem aktuellen Terminal können sie durch `source` aktiviert werden:

```
$ source ~/.bashrc
```

Im Folgenden werden die beiden Stellen in der Datei `~/.bashrc` gezeigt, die geändert werden müssen.

### Quellcode 1. Ursprüngliche ~/.bashrc

```
if [ "$color_prompt" = yes ]; then
    PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\0
33[01;34m\]\w\[\033[00m\]\$ ' ①
else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ ' ②
fi
```

① Setzt einen farbigen Prompt.

② Setzt einen monochromen Prompt.

### Quellcode 2. Geänderte ~/.bashrc

```
if [ "$color_prompt" = yes ] && false; then ①
    PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\0
33[01;34m\]\w\[\033[00m\]\$ '
else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
```

① Durch Einfügen von `&& false` wird stets der monochrome Prompt ausgewählt.

### Quellcode 3. Ursprüngliche ~/.bashrc

```
# enable color support of ls and also add handy aliases
if [ -x /usr/bin/dircolors ]; then
    test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors
-b)"
    alias ls='ls --color=auto' ①
    #alias dir='dir --color=auto'
    #alias vdir='vdir --color=auto'

    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi
```

① Hier wird ein Alias für eine farbige Ausgabe mit `ls` gesetzt.

*Quellcode 4. Geänderte ~/.bashrc*

```
# enable color support of ls and also add handy aliases
if [ -x /usr/bin/dircolors ] && false; then ①
    test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors
-b)"
    alias ls='ls --color=auto'
    #alias dir='dir --color=auto'
    #alias vdir='vdir --color=auto'

    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi
```

① Das Setzen der Aliasse wird durch Einfügen von `&& false` unterbunden.

*Quellcode 5. Ursprüngliche ~/.bashrc*

```
# If this is an xterm set the title to user@host:dir
case "$TERM" in
xterm*|rxvt*)
    PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1" ①
    ;;
*)
    ;;
esac
```

① Diese Anweisung ist für die Terminals in Ubuntu unnötig, führt aber dazu, dass bei Script der Prompt `user@host:dir` doppelt erscheint.

*Quellcode 6. Ursprüngliche ~/.bashrc*

```
# If this is an xterm set the title to user@host:dir
case "$TERM" in
xterm*|rxvt*)
#    PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1" ①
    ;;
*)
    ;;
esac
```

① In Shell-Scripts wird durch ein `#` am Zeilenanfang, diese Zeile „auskommentiert“ und dadurch nicht mehr ausgeführt. Der Prompt `user@host:dir` erscheint nun in den Aufzeichnungen mit Script nur noch einfach.

**Aufgabe 1**

Starten Sie die Aufzeichnung einer Shell-Session mit Script in einer Linux-VM und führen Sie die folgenden Arbeitsschritte durch:

1. Erstellen Sie den Ordner `script_docs` mit `mkdir`.
2. Legen Sie in diesem Ordner die Dateien `hello.adoc` und `ad.adoc` mit `touch` an.
3. Fügen Sie `"Hallo Script"` der Datei `hello.adoc` mit `echo "Hallo Script" >> script_docs/hello.adoc` an.
4. Fügen Sie der Datei `ad.adoc` die Zeile `"= AsciiDoctor"` an.
5. Lassen Sie mit `ls -l` den Inhalt des Ordners `script_docs` ausgeben und beenden Sie Script.

Überprüfen Sie anschließend mit `cat`, ob die Aufzeichnung erfolgreich war.

## 2. AsciiDoc

**AsciiDoc** ist eine leichtgewichtige Markup-Sprache, wie z. B. auch **Markdown**.

**AsciiDoctor** ist eine Ruby-Anwendung, die AsciiDoc-Dokumente in HTML5, PDF und andere Formate umwandelt.

Ein AsciiDoc-Dokument wird mit einem Texteditor wie GNU nano, gedit, emacs, vim, atom oder anderen verfasst bzw. bearbeitet — Textverarbeitungsprogramme wie Word oder Writer sind nicht geeignet. Anschließend werden die AsciiDoc-Dateien in ein Ausgabeformat übersetzt wie PDF, HTML5, Docbook oder andere.

### 2.1. Installation von AsciiDoctor

Für AsciiDoctor wird das Paket `ruby` benötigt:

```
$ sudo apt update
$ sudo apt install ruby
```

Mit dem Paket `ruby` wird auch **RubyGems**, das Paketsystem für Ruby installiert, dass zur Installation von `asciidoc` verwendet wird.

```
$ sudo gem install asciidoctor asciidoctor-diagram ①
$ sudo apt update
$ sudo apt install plantuml graphviz
$ mkdir srv
$ cd srv
$ nano index.adoc ②
$ asciidoctor -r asciidoctor-diagram -b html index.adoc ③
```

① Installation von AsciiDoctor und AsciiDoctor Diagram mit RubyGems.

② Erstellen eines AsciiDoc-Dokuments.

③ Mit `-r asciidoctor-diagram` wir die Erweiterung `Asciidoctor Diagram` verwendet.

## 2.2. Erste Schritte

Die wichtigsten Formatierungen werden anhand des folgenden Beispiels erläutert.

### Beispiel 3. Beispiel für ein AsciiDoc-Dokument

= Shell-Sessions mit AsciiDoctor dokumentieren ①

André Bauer ②

13. Mai 2018 ③

Shell-Sessions, die mit Script aufgezeichnet wurden, werden mit vier Strichen (Minuszeichen bzw. "dash") als Sourcecode-Block mit vorformatiertem Text eingerahmt.

④

Zusätzlich kann mit der Angabe `[source, sh]` die ⑤ Sprache angegeben werden. Hier ein Beispiel:

[source,sh] ⑥

---- ⑦

user@ubuntu-vb:~\$ cd srv/

user@ubuntu-vb:~/srv\$ ls

index.adoc index.html index.pdf inheritance.svg

user@ubuntu-vb:~/srv\$ touch diagramme.adoc ① ⑧

user@ubuntu-vb:~/srv\$ echo "= Diagramme mit AsciiDoctor" >> diagramme.adoc

user@ubuntu-vb:~/srv\$ cat diagramme.adoc

= Diagramme mit AsciiDoctor

----

<1> Erzeugt die Datei `diagramme.adoc`. ⑨

== Generieren einer HTML5-Datei ⑩

Ein AsciiDoc-Dokument wird dann mit dem folgenden Befehl in eine HTML-Datei umgewandelt.

[source,sh]

----

\$ asciidoctor -r asciidoctor-diagram -b html index.adoc

----

== Anzeigen mit Firefox

Mit `firefox index.html` wird diese im Browser angezeigt.

① Der Dokumententitel wird zu Beginn des Dokuments mit einem =-Zeichen angegeben.

② Die Angabe des Autors folgt in der Zeile nach dem Dokumententitel.

③ Das Datum wird direkt in der Zeile nach dem Autor angegeben.

④ Abschnitte werden durch Leerzeilen getrennt.

- ⑤ Mit Akzentzeichen (backticks) wird der Text für Kommandos oder Sourcecode-Ausschnitte in **Festbreitenschrift** gesetzt.
- ⑥ In den eckigen Klammern kann nach der Angabe **source** die verwendete Sprache, wie z. B. **java** oder **sh** für Shell, eingetragen werden. Dadurch wird die **Syntax farblich hervorgehoben**.
- ⑦ Shell-Befehle bzw. Source-Code-Blöcke werden mit vier Minuszeichen (dash) als vorformatierter Text gekennzeichnet bzw. eingerahmt.
- ⑧ Callout-Nummern werden im Source-Code-Block mit eckigen Klammern am Ende der Zeile eingetragen, d. h. **<1>**, **<2>** usw.
- ⑨ Callouts werden direkt nach dem Source-Code-Block erläutert.
- ⑩ Überschriften werden mit zwei oder mehreren **=**-Zeichen eingeleitet.

Das AsciiDoc-Dokument wird anschließend mit dem im Beispiel angegebenen Kommando übersetzt und kann dann im Browser angezeigt werden.

*Beispiel 4. Browserdarstellung eines mit AsciiDoctor generierten HTML5-Dokumentes*

## Shell-Sessions mit AsciiDoctor dokumentieren

André Bauer – 13. Mai 2018

Shell-Sessions, die mit Script aufgezeichnet wurden, werden mit vier Strichen (Minuszeichen bzw. "dash") als Sourcecode-Block mit vorformatiertem Text eingerahmt.

Zusätzlich kann mit der Angabe `[source, sh]` die Sprache angegeben werden. Hier ein Beispiel:

```
user@ubuntu-vb:~$ cd srv/
user@ubuntu-vb:~/srv$ ls
index.adoc index.html index.pdf inheritance.svg
user@ubuntu-vb:~/srv$ touch diagramme.adoc      (1)
user@ubuntu-vb:~/srv$ echo "= Diagramme mit AsciiDoctor" >> diagramme.adoc
user@ubuntu-vb:~/srv$ cat diagramme.adoc
= Diagramme mit AsciiDoctor
```

1. Erzeugt die Datei `diagramme.adoc` .

## Generieren einer HTML5-Datei

Ein AsciiDoc-Dokument wird dann mit dem folgenden Befehl in eine HTML-Datei umgewandelt.

```
$ asciidoctor -r asciidoctor-diagram -b html index.adoc
```

## Anzeigen mit Firefox

Mit `firefox index.html` wird diese im Browser angezeigt.



Weitere Informationen, wie man ein AsciiDoc-Dokument gestalten kann, stehen im [AsciiDoc Writer's Guide](#) und im [Asciidoctor User Manual](#).

## Aufgabe 2

Erstellen Sie aus der aufgezeichneten Shell-Session aus Aufgabe 1 ein AsciiDoc-Dokument, indem Sie Titelangaben und Erläuterungen hinzufügen sowie die Shell-Kommandos als Sourcecode-Blöcke kennzeichnen und Callouts einfügen. Generieren Sie mit Asciidoctor aus der AsciiDoc-Datei eine HTML5-Datei und überprüfen Sie das Ergebnis im Browser.

## 2.3. Diagramme mit Asciidoctor erstellen


Über die Erweiterung [Asciidoctor Diagram](#) können Diagram-Beschreibungen direkt in AsciiDoc-Dokumente eingebettet werden. Asciidoctor sorgt dann automatisch für die Übersetzung und fügt die erzeugten Grafiken dann als Bilder ein. Voraussetzung ist, dass die dazu notwendigen Programme installiert sind.

### 2.3.1. PlantUML

[PlantUML](#) ist ein Werkzeug, um (UML-)Diagramme zu erstellen. Über die Erweiterung [Asciidoctor Diagram](#) kann man PlantUML-Diagramme direkt in AsciiDoc-Dokumenten beschreiben. Diese werden in [PNG-Grafiken](#) oder [SVG-Grafiken](#) übersetzt.

Für PlantUML gibt es u. a. Plug-Ins für [Eclipse](#) und [Jet Brains IDEs wie IntelliJ IDEA](#).

*Beispiel 5. Beispiel für die Verwendung von PlantUML in einem AsciiDoc-Dokument und Ausgabe im Browser*

AsciiDoc	Darstellung im Browser
<pre> == Assoziation zwischen den Klassen A und B  [plantuml, class-association, svg] ① .... ② class A  class B  A -up-&gt; B .... </pre> <p>① In den eckigen Klammern wird das Diagrammwerkzeug, der Dateiname und das Zielformat festgelegt. Hier wird die Diagramm-Beschreibung mit PlantUML in eine SVG-Grafik übersetzt und in der Datei <code>class-association.svg</code> gespeichert.</p> <p>② Diagramm-Beschreibungen werden als Block mit vier Punkten eingerahmt.</p>	<h3>Assoziation zwischen den Klassen A und B</h3> 

Die Diagramm-Beschreibungs-Sprache von PlantUML ist anhand zahlreicher Beispiel unter [PlantUML.com](https://plantuml.com) sowie im [PlantUML Language Reference Guide](#) dokumentiert.

### 2.3.2. blockdiag, actdiag, nwdiag und seqdiag

Die Programme `blockdiag`, `seqdiag`, `actdiag` und `nwdiag` übersetzen -- ähnlich wie PlantUML -- Diagramm-Beschreibungen in Grafiken.

Sie können unter Ubuntu über das Paketmanagement-System installiert werden:

```

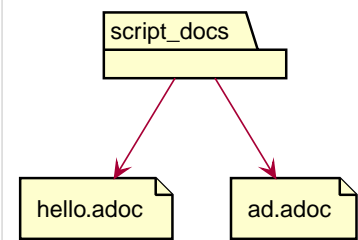
$ sudo apt update
$ sudo apt install python-blockdiag python-actdiag python-nwdiag python-seqdiag

```

## Aufgabe 3

Fügen Sie Ihrem AsciiDoc-Dokument aus den Aufgaben 1 und 2 eine Diagramm-Beschreibung für die nebenstehende Grafik mit PlantUML hinzu und lassen Sie AsciiDoctor daraus ein HTML5-Dokument mit SVG-Grafik erstellen.

Hilfe zum Erstellen einer solchen Grafik finden Sie unter <http://plantuml.com/deployment-diagram>.



## 2.4. PDF-Dokumente generieren

PDF-Dokumente werden aus AsciiDoc-Dokumenten mit der Erweiterung **AsciiDoctor PDF** durch Angabe von **-r asciidoctor-pdf** und des Ausgabeformats mit **-b pdf** generiert.

### 2.4.1. Installation

```
$ sudo gem install asciidoctor-pdf --pre
```

### 2.4.2. Anwendung

```
$ asciidoctor -r asciidoctor-diagram -r asciidoctor-pdf -b pdf -a allow-uri-read index.adoc
```

Für PDF-Dokumente gibt es unter Linux verschiedenen Viewer wie **evince** oder **atril**, die die Dokumente nach Änderungen automatisch neu laden.

```
$ evince index.pdf &
```

## Aufgabe 4

Generieren Sie aus Ihrem AsciiDoc-Dokument aus den Aufgaben 1-3 eine PDF-Datei, die Sie in Ihr E-Portfolio einbinden.