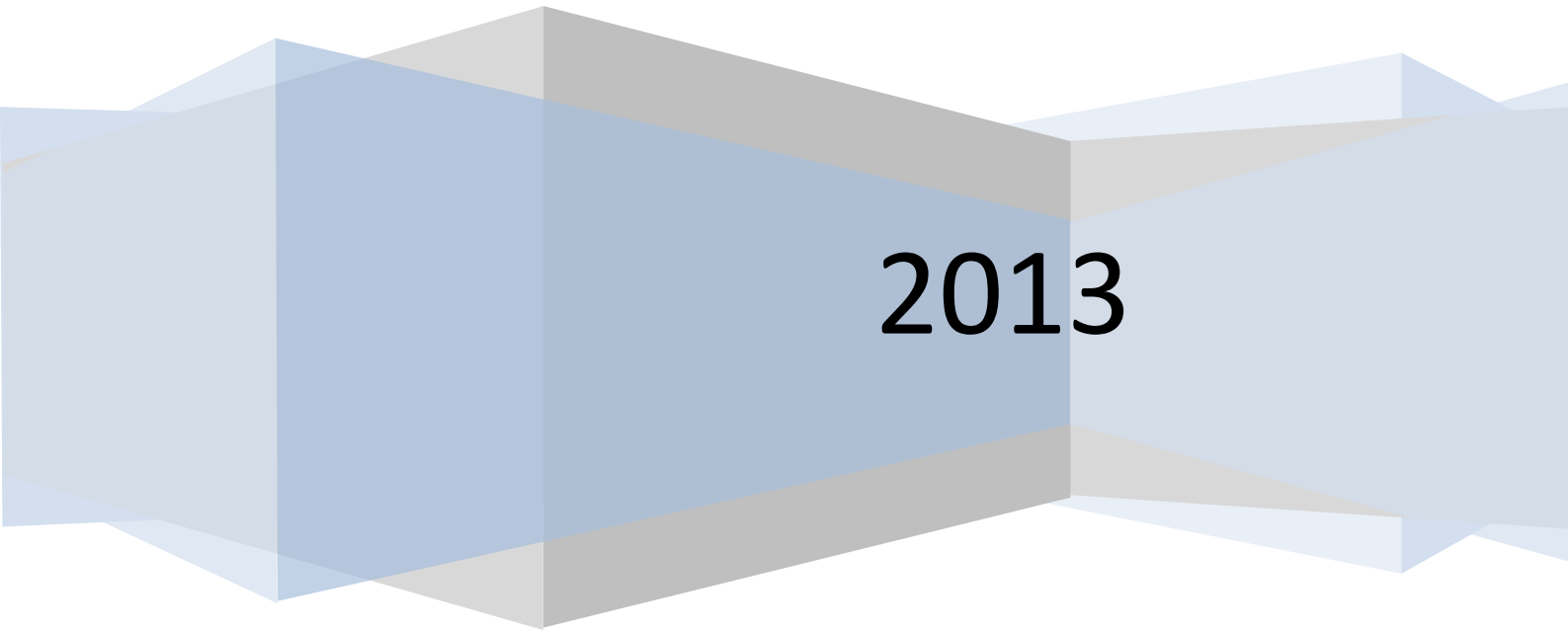
myMLD Dokumentation

zur Automatisierung der MLD-Entwicklungsumgebung

skippy



2013

Inhalt

Inhaltsverzeichnis

Motivation

Beim Neuaufsetzen meiner verschiedenen MLDs wusste ich nicht mehr, was ich konkret an Konfigurationsänderungen vorgenommen hatte. Zwar hatte ich mal angefangen, eine kurze Dokumentation anzulegen, aber die war schnell veraltet bzw. unvollständig. Inspiriert durch einen Artikel im Forum über die Config-Addons, habe ich damit etwas experimentiert und gute Ergebnisse erzielt. Dann kam der Wunsch auf, die Erstellung meiner ISOs mit den Konfigurationen zu automatisieren. Zwangsläufig musste ich mich dazu mit der Shell-Programmierung beschäftigen. Dabei entstanden immer wieder neue Ideen, was noch automatisiert werden könnte, um mir monotone Tipparbeit zu ersparen.

Letztendlich ist jetzt ein kleines Programmpaket entstanden, das die Einrichtung der Entwicklungsumgebung, das Aktualisieren der Addons und das Erstellen meiner ISOs übernimmt. Dieses Paket möchte ich euch auch gern zur Verfügung stellen. Es ist noch nicht vollständig ausgetestet und enthält sicher noch ein paar Fehler.

Installation der Verzeichnisse und Dateien von myMLD

Die Verzeichnisse und Dateien sind in einem TAR-File gepackt. Die Datei heißt myMLD.tar und wird mit dem Befehl

```
tar xvf myMLD.tar
```

im aktuellen Verzeichnis entpackt. Alternativ kann die Datei auch mit dem Befehl

```
tar xvf myMLD.tar -C <Zielverzeichnis>
```

in einem beliebigen Verzeichnis entpackt werden. Nach dem Entpacken befindet sich in dem Zielverzeichnis ein Verzeichnis mit dem Namen myMLD. Darin sind Programme, diese Datei sowie weitere Verzeichnisse und Dateien vorhanden, auf die nachfolgend eingegangen wird.

Die Konfigurationsdatei „myMLD.cfg“

Diese Datei befindet sich im Verzeichnis myMLD und muss auch dort belassen werden, weil sie sonst von den nutzenden Programmen nicht gefunden wird. Hier sind die Variablen definiert, die für die Verarbeitung benötigt werden. Die gesetzten Werte sind Standardwerte und können bei Bedarf an die jeweiligen Gegebenheiten angepasst werden. Eine Auflistung der verwendeten Variablen mit deren Bedeutung ist im Anhang beigefügt.

Das Programm „build“

Mit diesem Programm kann eine Entwicklungsumgebung für die MLD aufgebaut werden. Diese Entwicklungsumgebung kann aktualisiert und kompiliert werden. Weiterhin können eigene ISOs erstellt und der Develserver gestartet und gestoppt werden. Die Funktionen sind über Menüs (mit dem Shell-Befehl dialog realisiert) auswählbar.

Zu Beginn wird geprüft, ob die eingestellte MLD-Version in der Datei myMLD.cfg zum installierten Ubuntu-System passt. Die Informationen werden aus der Datei mld_ubuntu.dat entnommen. Sollte dies nicht passen, wird das Programm mit einer entsprechenden Meldung abgebrochen.

Hinweis: Momentan wird von der MLD nur die 32-Bit-Version von Ubuntu unterstützt.

Im Titel des Hauptmenüs wird die in der Datei myMLD.cfg eingestellte MLD-Version angezeigt. Sollte dies nicht die gewünschte MLD-Version sein, so ist das Programm zu beenden und die gewünschte MLD-Version in der Datei zu korrigieren.

Nachfolgend werden die einzelnen Menüpunkte erläutert.

MLD installieren

Welche Schritte zur Installation notwendig sind, ist ausführlich im dem dazugehörigen Wiki-Artikeln http://www.vdr-wiki.de/wiki/index.php/MLD#MLD_3 erläutert. Diese darin beschriebenen Schritte habe ich in dem Programm automatisiert. Bevor mit der Installation begonnen wird, erfolgt eine Prüfung, ob das MLD Verzeichnis noch nicht vorhanden ist. Wenn die Prüfungen positiv verlaufen sind, erfolgt die Abfrage, ob nach dem Checkout auch alles kompiliert werden soll. Weiterhin wird das Passwort der Benutzerkennung abgefragt, weil u.a. für die Paketinstallationen sudo-Rechte benötigt werden.

Nach Abschluss der Installation sollte ein Blick in das Logfile geworfen werden. Wurde die Frage nach der Kompilierung positiv bestätigt, sollte geschaut werden, ob die vordefinierten ISOs, aus der Datei MLD_DIR/Makefile unter addon_lists auch im Verzeichnis $PAKET_DIR vorhanden sind. Wenn nicht, dann hat es Fehler gegeben, die analysiert werden sollten.

Ein Abbruch erfolgt, wenn das MLD-Verzeichnis bereits vorhanden ist.

Abweichend von dem Wiki-Artikel wird das Paket lirc nicht über die Abhängigkeiten, sondern bereits am Anfang installiert. Der Grund dafür ist, dass bei lirc noch zwei Konfigurationsbestätigungen durch den Benutzer/der Benutzerin erfolgen müssen (jeweils none auswählen und ok). Hiermit wird versucht, die manuelle Bestätigung so schnell wie möglich erfolgen zu lassen, so dass die sehr lange dauernde Installation dann ohne weitere Eingriffe erfolgen kann (je nach System ist eine Dauer von mehreren Stunden einzuplanen). Folgende Menüpunkte können gewählt werden:

MLD aktualisieren

Es wird gefragt, ob sich nach der Aktualisierung eine Übersetzung der Sourcen anschließen soll. Weiterhin wird das Benutzerpasswort abgefragt, das zur Auflösung und Installation der abhängigen

Pakete benötigt wird. Danach wird entsprechend der Ausgabe von „svn up“ im $MLD_DIR ein checkout_all oder update_all durchgeführt, ggf. neue abhängige Pakete heruntergeladen und installiert und wenn gewünscht die Kompilierung der Sourcen durchgeführt.

Auch hier empfiehlt sich, nach dem Ende ein Blick in das Logfile und bei durchgeführter Kompilierung in das $PAKET_DIR zu werfen. Hinweis: Sollten inzwischen eigene ISOs erstellt werden, so wird nur das ISO neu erstellt, das in $MLD_DIR/Makefile.config eingetragen ist. Mehr dazu siehe Kapitel eigene ISOs erstellen.

Ist die MLD noch nicht installiert (es wird geprüft, ob das MLD_DIR vorhanden ist), erfolgt eine Fehlermeldung und es wird wieder ins Hauptmenü verzweigt.

MLD kompilieren

Es werden die Abhängigkeiten aufgelöst und ggf. neue Pakete installiert. Dazu wird das Benutzerpasswort abgefragt. Danach wird ein make all durchgeführt, also die veränderten Sourcen neu kompiliert und das/die ISOs erstellt. Wie bereits zuvor erwähnt, ist auch hier nach Fertigstellung der Blick ins Logfile und $PAKET_DIR sinnvoll.

Auch hier wird zunächst geprüft, ob die MLD installiert ist.

ISO erstellen

Damit wird das Erstellen eines eigenen ISO durchgeführt. Voraussetzung ist natürlich, dass die MLD-Entwicklungsumgebung installiert ist und die benötigten Verzeichnisse und Dateien unter $myMLD_DIR vorhanden sind.

Ein ggf. vorhandenes Config-Addon im Verzeichnis $MLD_DIR mit dem Namen config.myaddon wird gelöscht und neu angelegt. Danach werden die benötigten Dateien aus dem Verzeichnis $myMLD_DIR in das neue Config-Addon kopiert. Es besteht die Möglichkeit eine bestehende Aufnahme mit in das ISO zu packen, um z. B. die Funktionalitäten des VDR in einer VM ohne TV-Empfang testen zu können. Danach wird das ISO im Verzeichnis $PAKET_DIR erstellt und sofern die Variable gesetzt ist, in den gemeinsamen Ordner kopiert.

Da das Erstellen eigener ISOs etwas umfangreicher ist, habe ich dafür ein eigenes Kapitel im Anhang eingerichtet => eigene ISOs erstellen.

Develserver

Hinter diesem Menüpunkt erscheint ein weiteres Menü mit den Punkten starten und beenden. Was sich hinter dem Develserver verbirgt, ist in dem Wiki Artikel http://www.vdr-wiki.de/wiki/index.php/MLD_3.0_-_Entwicklungsumgebung_als_Updateserver_verwenden beschrieben. Bei Punkt 1 wird das Addon develserver ausgecheckt, sofern es noch nicht vorhanden ist und mit make gestartet. Unter Punkt 2 wird ein make clean ausgeführt und damit der Develserver beendet. Mit dem Menüpunkt zurück wird wieder in das Hauptmenü verzweigt. Das Untermenü wird nur dann angezeigt, wenn auch die MLD-Entwicklungsumgebung im Verzeichnis $MLD_DIR installiert ist.

Programm beenden
Sollte selbsterklärend sein ;-).

Das Programm „pack“
Dies ist ein kleines Programm, mit dem die Verzeichnisse und Dateien von myMLD gepackt und entpackt werden können. Nach dem Start wird zunächst geprüft, ob die Datei myMLD.cfg vorhanden ist und das dort in der Variable $myMLD_DIR definierte Verzeichnis vorhanden ist. Wenn nicht, wird das Programm mit einer entsprechenden Meldung beendet. Ist alles ok, erscheint ein Menü mit folgenden Auswahlmöglichkeiten:

TAR-Datei wird erstellt (ohne Film)
Das Verzeichnis $myMLD_DIR wird gepackt und im Verzeichnis $GEMEINSAMER_ORDNER mit dem Namen myMLD.tar abgelegt. Nicht in das TAR-File kommen Dateien, die mit „.svn“ beginnen oder mit „~“ bzw. „.log“ enden. Ebenfalls werden alle Dateien ausgeschlossen, die im Verzeichnis $myMLD_DIR/all/tv liegen. Dieses Verzeichnis ist für die Ablage von VDR-Aufnahmen vorgesehen.

TAR-Datei wird erstellt (mit Film)
Wie vor, jedoch werden hier auch die Dateien unter $myMLD_DIR/all/tv mit in das File übernommen.

TAR-Datei wird entpackt
Die Datei myMLD.tar im Verzeichnis $GEMEINSAMER_ORDNER wird im Verzeichnis $myMLD_DIR entpackt. Bestehende Dateien werden ohne Rückfrage überschrieben.

In die SVN_Workarea kopieren
Zunächst wird die TAR-Datei wie unter „TAR-Datei wird erstellt (ohne Film)“ beschrieben, erstellt. Danach wird diese Datei im Verzeichnis $SVN_DIR wieder entpackt. Dieser Menüpunkt dient dazu, die Dateien in die Workarea des KM-Tools zu kopieren.

Programm beenden
Siehe oben.

Anhang

eigene ISOs erstellen

Mit myMLD ist es möglich, sich seine maßgeschneiderte MLD-Installation zu erstellen. Einerseits kann festgelegt werden, welche Pakete sich im ISO befinden sollen und andererseits bereits im Vorfeld über ein eigenes Config-Addon die Anpassungen an die Fernbedienung, die benötigte Kanalliste und die bevorzugten VDR-Parameter eingestellt werden. Im Idealfall hat man dann „sein“ Wunschsystem beim Booten direkt auf dem Medium.

Grundsätzlicher Aufbau

Zur ISO-Erstellung gibt es im Verzeichnis $myMLD das Verzeichnis „all“. Als Unterverzeichnisse sind die MLD-Versionen abgebildet. Darin werden Verzeichnisse und Dateien abgelegt, die für alle ISOs einer MLD-Version gleich sind. Dies ist z.B. das Verzeichnis „usr“, das die für das Config-Addon die notwendigen Dateien zur Dokumentation (handling, readme und settings) enthält, das Verzeichnis „channels“, das die Kanalliste(n) beinhaltet und das Verzeichnis „tv“ für eigene Aufzeichnungen, die man z.B. zu Testzwecken im ISO haben möchte. Es können bei Bedarf auch weitere Verzeichnisse angelegt werden, um z.B. Konfigurationsdateien der VDR-Plugins o.ä. zu übernehmen.

Weiterhin gibt es für jedes der 5 möglichen ISOs, die mit myMLD erstellt werden können, ein Verzeichnis, das die dafür spezifischen Dateien enthält. Auch hier sind die MLD-Versionen als Unterverzeichnisse abgebildet. Zwingend darin enthalten sein müssen die Datei „config.sh“, die die Befehle zum Aufbau des Config-Addons beeinhaltet, die Datei „INSTALL“, in der die Befehle für die Installation des Paketes enthalten sind und die Datei „Makefile.config“. Darin ist festgelegt, welche Pakete in das jeweilige ISO gepackt werden. Das Verzeichnis kann auch weitere Dateien enthalten, die für das jeweilige ISO benötigt werden. Dies können z. B. die remote.conf und die lirc.conf sein, die bereits für die eigene Fernbedienung konfiguriert sind. Nachfolgend versuche ich diese Infos mit einigen Ergänzungen in Tabellenform noch etwas übersichtlicher darzustellen:

Name	Beschreibung
$myMLD/all/$MLD_VERSION	Verzeichnis, das die Unterverzeichnisse und Dateien enthält, die bei allen eigenen ISOs identisch sind.
$myMLD/all/$MLD_VERSION/avahi	Ich bin absoluter Fan vom avahi-linker, den es ab MLD-Version 3.0.3 gibt. Hier habe ich die auf meine Bedürfnisse angepassten Konfigurationsdateien abgelegt.
$myMLD/all/ $MLD_VERSION/channels	Verzeichnis, das die Kanalliste enthält. Die Kanalliste ist normalerweise in einem Haushalt für alle VDRs identisch. Ich habe jedoch Installationen, die HD empfangen und anzeigen können, als auch ältere Geräte, die nur mit SD umgehen können. Deshalb habe ich 2 Kanallisten in diesem Verzeichnis.
$myMLD/all/$MLD_VERSION/plugins	Dieses Verzeichnis ist für die selbst veränderten Konfigurationsdateien der VDR-Plugins gedacht, sofern sie nicht über Parameter der setup.conf verändert werden – z.B. svdrphosts.conf.

Name	Beschreibung
$myMLD/all/$MLD_VERSION/tv	Hier kann eine – möglichst kleine – VDR-Aufzeichung abgelegt werden, um diese z. B. für Testzwecke mit in das ISO aufzunehmen. Ich nutze es bei Installationen in einer VM-Ware, die selbst keine TV-Karte für Aufzeichnungen hat.
$myMLD/all/$MLD_VERSION/usr	In den Unterverzeichnissen share/doc/config befinden sich die drei Dateien handling, readme und settings. Sie müssen bei jedem Paket der MLD dabei sein. Wer möchte, kann sie auch mit sinnvollen Inhalten für seine Config-Addons füllen. Da es sich jedoch um ein eigenes Addon handelt, das keine Verbreitung finden wird, können die Dateien auch so belassen werden.
$myMLD/$VDR{1-5}[1]/$MLD_VERSION	Hier werden die für das ISO $VDR{1-5}[1] gespeicherten Dateien gespeichert. Es müssen mindestens die Dateien config.sh, INSTALL und Makefile.config vorhanden sein, die nachfolgend erklärt werden.
$myMLD/$VDR{1-5}[1]/ $MLD_VERSION/config.sh	Diese Datei kopiert die für das ISO benötigten Dateien aus $myMLD an die richtige Stelle im Config-Addon und erzeugt ggf. benötigte Verzeichnisse. Sie wird vom Programm build aufgerufen und erhält die Übergabeparameter $MLD_DIR, $myMLD_DIR und $VDR{1-5}[1]. Diese Datei ist an die eigenen Bedürfnisse anzupassen!
$myMLD/$VDR{1-5}[1]/ $MLD_VERSION/INSTALL	Diese Datei wird bei der Installation des Config-Addons auf dem Zielsystem ausgeführt. Mit ihr können z. B. die VDR-Parameter in der setup.conf und die rc.config (Konfigurtationsdatei der MLD) auf die eigenen Bedürfnisse angepasst werden. Diese Datei ist an die eigenen Bedürfnisse anzupassen!
$myMLD/$VDR{1-5}[1]/ $MLD_VERSION/Makefile.config	In dieser Datei sind die Pakete (Addons und VDR-Plugins) aufgelistet, die in das ISO gepackt und auf dem Zielsystem installiert werden. Die Datei wird vom Kommando make verarbeitet. Diese Datei ist an die eigenen Bedürfnisse anzupassen! **Wichtig:** Der darin verwendete Name für das ISO (siehe Zeile 4 im Kapitel Makefile.conf (Muster)) muss auch vor dem „=“ enthalten sein, wie er in der Datei myMLD.cfg definiert wurde. Beispiel: $VDR1=VDR1. Der Name des ISOs darf „myVDR1“ oder nur „VDR1“ sein. Entscheidend ist, dass „VDR1“ darin vor kommt.

Nachfolgend habe ich Muster der vorgenannten Dateien config.sh, INSTALL und Makefile.conf beigefügt und versucht, diese zu kurz erläutern. Die Inhalte sind aus den Dateien von meinem MLD-HD-Client entnommen und sicher noch nicht vollständig (z.B. fehlt noch die angepasste menu.xml, die die Menüstruktur des VDR abbildet). Die Dateien sind unbedingt auf eure eigenen Installationen

[1] bezeichnet den Namen des ISO, wie in der Datei myMLD.cfg definiert.

anzupassen. Mit etwas Aufwand ist es möglich, die Config-Addons so aufzubauen, dass ihr später eure MLD Installationen voll automatisiert erstellen lassen könnt.

Da alle Dateien in dem Verzeichnis $myMLD_DIR gespeichert sind, gehen sie auch nicht verloren, wenn ihr auf eine andere Version der MLD-Entwicklungsumgebung wechselt. Mit dem Programm pack wird das Verzeichnis gepackt und in eurem gemeinsamen Ordner abgelegt. In der neuen Entwicklungsumgebung kann es dann wieder ausgepackt werden.

weitere Vereinfachung durch Links

Wie zuvor beschrieben, greift das Programm build auf die vorgenannten Verzeichnisse zu, um das ISO zu erstellen. Jedoch ist es häufig der Fall, dass Dateien auch dann gleich sind, wenn sich MLD-Version und/oder das ISO unterscheiden. Damit diese gleichen Verzeichnisse und Dateien nun nicht an verschiedenen Stellen redundant gespeichert werden müssen, können diese Verzeichnisse auch jeweils auf der darüber liegenden Ebene angelegt und verlinkt werden. Ich versuche es mal am Beispiel der Aufnahme zu verdeutlichen:

Eine VDR-Aufnahme, die in ein ISO kopiert werden soll, ist für alle ISO‘s identisch. Normalerweise müsste sie jeweils in die Verzeichnisse „$myMLD_DIR/all/3.0.0/tv“, „$myMLD_DIR/all/3.0.1/tv“, „$myMLD_DIR/all/3.0.1.1/tv“, „$myMLD_DIR/all/3.0.2/tv“ und „$myMLD_DIR/all/3.0.3/tv“ kopiert werden, wenn man für all diese Versionen ISOs mit einer Aufnahme erstellen möchte. Deshalb kann das Verzeichnis „tv“ mit der Aufnahme eine Ebene höher, also unter $myMLD_DIR/all gespeichert werden. Für das Verzeichnis $myMLD_DIR/all/$MLD-VERSION/tv wird ein Softlink mit absolutem Pfadnamen[2] auf das übergeordnete Verzeichnis gelegt.

Beispiel: (natürlich müssen die nachstehenden Variablennamen entsprechend den eigenen Gegebenheiten aufgelöst werden, da die Shell sie nicht kennt)

```
mv $myMLD_DIR/all/$MLD_VERSION/tv $myMLD_DIR/all
ln -s $myMLD_DIR/all/tv/ $myMLD_DIR/all/$MLD_VERSION/tv
```

Dieses Beispiel kann nun auf alle Verzeichnisse analog angewandt werden. Bei Dateien kann auch ein Softlink mit relativen Pfadnamen verwendet werden. Nachfolgend das Beispiel mit der Datei Makefile:

```
cd $myMLD_DIR/all/$MLD_VERSION
mv Makefile ..
ln -s ../Makefile Makefile
```

Dieses Vorgehen bietet den Vorteil, dass durch die einmalige Speicherung Plattenplatz gespart wird. Der große Vorteil bei den Dateien ist jedoch, dass diese bei Änderungen nur einmal bearbeitet werden müssen. Dadurch ist die Wahrscheinlichkeit geringer, dass sich Fehler einschleichen.

[2] Bei Verzeichnissen ist ein Softlink mit absolutem Pfadnamen notwendig, weil beim Kopieren durch das Programm build sonst nur der Link und nicht die eigentlichen Verzeichnisse und Dateien kopiert werden.

config.sh (Muster)

```
1. #!/bin/sh
2.
3. MLD_DIR=$1
4. myMLD_DIR=$2
5. VDR=$3
6.
7. # Kanalliste kopieren
8. mkdir -p $MLD_DIR/config.myaddon/template/etc/vdr/channels
9. cp $myMLD_DIR/all/channels/my_channels_hd.conf
   $MLD_DIR/config.myaddon/template/etc/vdr/channels
10. # Dateien für Fernbedienung kopieren
11. cp $myMLD_DIR/$VDR/lircd.conf $MLD_DIR/config.myaddon/template/etc
12. cp $myMLD_DIR/$VDR/remote.conf $MLD_DIR/config.myaddon/template/etc/vdr
13.
14. return 0
```

Erläuterungen:

Zeile 1	Dieses Programm wird mit der Shell (/bin/sh) ausgeführt
Zeile 2	Leerzeile zur besseren Übersicht
Zeilen 3 – 5	die Übergabeparmeter erhalten einen sprechenden Variablennamen.
Zeile 6	Leerzeile zur besseren Übersicht
Zeile 7	Kommentarzeile zur Erläuterung
Zeile 8	das benötigte Verzeichnis zur Speicherung der Kanalliste im Config-Addon wird angelegt.
Zeile 9	die individuelle Kanalliste wird in das Config-Addon kopiert.
Zeile 10	Kommentarzeile zur Erläuterung
Zeilen 11 + 12	die Konfigurationsdateien für die Fernbedienung werden in das Config-Addon kopiert.
Zeile 13	Leerzeile zur besseren Übersicht
Zeile 14	Rückgabewert 0 => Programm wurde ordnungsgemäß beendet

INSTALL (Muster)

```
#!/bin/sh
case "$1" in
       install)
              echo "Mounten der Datenplatte" >> /var/log/sysinit
              mount /dev/sdd1 /mnt/data

              . /etc/init.d/rc.functions

              echo "Ändern der rc.config" >> /var/log/sysinit
              update_setting "HOST_NAME" "VDR1"
              update_setting "ALSA_DEVICE" "1 7 HDMI 1"
              update_setting "NETWORK_WOL" "1" "Aktivate wakeup on lan  (1=on, 0=off)"
              update_setting "VDR_CHANNELLIST" "my_channels_hd"
              update_setting "VDR_PLUGIN_ARGS_graphlcd" "-c /etc/graphlcd.conf -d ax206dpf -s pearldpf-simple"
              update_setting "LIRC_ARGS" "/dev/ttyS0 uart none"
              update_setting "LIRC_DRIVER" "lirc_serial"
              update_setting "NETWORK_SERVER_IP" "xxx.xxx.xxx.xxx"
              update_setting "NETWORK_SERVER_MAC" "xx:xx:xx:xx:xx:xx"
              update_setting "NETWORK_SERVER_WAKEUP" "1"
              update_setting "NETWORK_SERVER_SHUTDOWN" "1"
              update_setting "BOOTMENU_TIME" "1"

              # BOOTMENU_TIME ins Bootmenü schreiben
              sed "s/TIMEOUT .*/TIMEOUT $(($BOOTMENU_TIME*10))/" -i /boot/isolinux/style.cfg

              echo "Ändern der setup.conf" >> /var/log/sysinit
              cat >> /etc/vdr/setup.conf.add <<- EOF
                     FontFixSizeP = 0.028
                     FontOsdSizeP = 0.030
                     FontSmlSizeP = 0.025
                     svdrpservice.ServerIP = xx.xxx.xxx.xxx
                     svdrpservice.ServerPort = 6419
                     streamdev-client.RemoteIP = xx.xxx.xxx.xxx
                     streamdev-client.HideMenuEntry = 1
              EOF
              ;;
       uninstall)
              ;;
       depend)
              ;;
       *)
              echo "Usage $0 {install | uninstall | depend}" >&2
              exit 1
esac
```

Erläuterungen:

Zeile 1	Dieses Programm wird mit der Shell (/bin/sh) ausgeführt.
Zeile 2	Mehrfachauswahl mit dem 1. Übergabeparameter wird eingeleitet.
Zeile 3	wenn 1. Übergabeparameter = install ist, dann werden die Zeilen 4 – 36 ausgeführt.
Zeile 4	Ausgabe des Strings in der Logdatei sysinit.
Zeile 5	Die Datenpartition wird eingebunden.
Zeile 6	Leerzeile zur besseren Übersicht
Zeile 7	Die Datei rc.functions wird ausgeführt. Dadurch erhält das Programm Zugriff auf die darin gesetzten Variablen und kann die dort definierten Funktionen (z.B. update_setting) nutzen.
Zeile 8	Leerzeile zur besseren Übersicht
Zeile 9	Ausgabe des Strings in der Logdatei sysinit.
Zeilen 10 – 21	Die dort genannten Parameter werden in der /etc/rc.config auf dem Zielsystem entsprechend gesetzt bzw. verändert.
Zeile 22	Leerzeile zur besseren Übersicht
Zeile 23	Kommentarzeile mit Erläuterungen.
Zeile 24	In Zeile 21 wurde zwar die Zeit für die Dauer der Anzeige des Bootmenüs geändert. Der Wert muss jedoch noch in die Datei style.cfg geschrieben werden. Dies wird hierdurch erledigt.
Zeile 25	Leerzeile zur besseren Übersicht
Zeile 26	Ausgabe des Strings in der Logdatei sysinit.
Zeilen 27 – 35	Es werden die Inhalte der Zeilen 29 – 34 in die Datei setup.conf.add am Ende angehängt. Sollte die Datei noch nicht vorhanden sein, so wird sie angelegt. Beim Startvorgang der MLD wird geschaut, ob die Datei setup.conf.add vorhanden ist. Wenn ja, werden vor dem Start des VDR die darin enthaltenen Parameter in die Konfigurationsdatei des VDR (setup.conf) geschrieben bzw. dort verändert.
Zeile 36	Beendet die Mehrfachauswahl „install“.
Zeilen 37 + 38	Einsprung und Ende, wenn der 1. Übergabeparameter = unistall ist.
Zeilen 39 + 40	Einsprung und Ende, wenn der 1. Übergabeparameter = depend ist.
Zeile 41	Einsprung, wenn keine zuvor definierte Auswahl zutreffend war.
Zeile 42	Ausgabe des Stings auf dem stderr Kanal.
Zeile 43	Beenden des Programms mit Fehlerrückgabe.
Zeile 44	Beendet die in Zeile 2 eingeleitete Mehrfachauswahl.

Makefile.conf (Muster)

```
1. .SILENT:
2. # Eigene ISO's
3. addon_lists ?="\
4. myVDR1 = acpi addons alsa autofs avahi avahi-linker backup bash
   config.myaddon control dbus eventlircd extrecmenu fuse-utils graphlcd
   graphlcd-base install irkeytable lib-av lib-curl lib-pango lib-x lirc
   locales mhddfs network nfs-server nss-mdns pearldpf-simple perl pin
   python python-avahi python-dbus remoteosd remotetimers sensors setup
   softhddevice ssh streamdev-client svdrpservice vdr webserver xorg xorg-
   nvidia yaepghd\n\
5. "
```

Erläuterungen:

Die Datei Makefile.conf übersteuert die im $MLD_DIR/Makefile gesetzten Parameter. So werden nun nicht die im Makefile definierten ISOs erstellt, sondern nur das in dieser Datei definierte. Natürlich könnte die Änderung auch im Makefile erfolgen, wäre dann aber nach einem Update der Datei verloren.

Zeile 1	Die automatische Ausgabe von Meldungen durch make wird ausgeschaltet
Zeile 2	Kommentarzeile zur Erläuterung
Zeile 3	Eröffnet den „Container“, der die ISOs mit den Paketen enthält, die erzeugt werden sollen. Über myMLD wird immer genau nur 1 ISO definiert.
Zeile 4	Enthält den Namen der im ISO verwendet wird (VDR1) und benennt alle Pakete, die in das ISO gepackt werden sollen. Wichtig ist „\n\“, der das Ende für dieses ISO markiert.
Zeile 5	beendet den in Zeile 3 geöffneten „Container“

definierte Variablen in myMLD.cfg

Name	Beschreibung
GEMEINSAMER_ORDNER	Verzeichnis des gemeinsamen Ordners der virtuellen Maschine (Gastsystem) und dem Host zum Datenaustausch. Wird vom Programm pack verwendet, um dieses Programmpaket dort als TAR-File zu speichern oder ein dort liegendes TAR-File im myMLD-DIR zu entpacken. Ebenfalls genutzt vom Programm build, um die ISOs nach Erstellung in diesen Ordner zu kopieren. Sollte kein gemeinsamer Ordner vorhanden sein, sollte die Variable Standardwert: /media/sf_VirtualBox-share
LOGFILE	Pfad und Dateiname, in dem die Log-Einträge des Programms build gespeichert werden. Das Logfile dient zur Überprüfung des korrekten Programmablaufs und soll bei der Fehleranalyse helfen. Standardwert: $myMLD_DIR/build.log
MLD_CHECKOUT	MLD_CHECKOUT="TRUE" => die MLD-Sourcen werden aus Subversion via Checkout geholt. MLD_CHECKOUT="FALSE" => Ab der MLD 3.0.3 werden die kompletten MLD-Sourcen auch in einem TAR-File zum Download[3] bereit gestellt. Wenn dieser Weg beschritten werden soll, z.B. weil der Download des Checkout immer wieder wegen Kapazitätsproblemen beim Internetzugang zusammen bricht, muss derzeit diese Datei noch manuell in das Verzeichnis ~/Downloads heruntergeladen werden. Sie wird beim Parameter „FALSE" vom Programm build dort erwartet und verarbeitet. Ein automatischer Download steht auf der todo-Liste. Weitere derzeitige Einschränkung: Die Datei darf nur 1x im Download-Verzeichnis vorhanden sein (Suchmuster: MLD-3.0.3_all-sources*.tgz) Standardwert: TRUE
MLD_DIR	Verzeichnis, in dem die MLD-Entwicklungsumgebung gespeichert wird. Die Variable ist in der Datei myMLD.cfg mit dem Verzeichnisnamen MLD im Homeverzeichnis des Benutzers vorbelegt. Standardwert: ~/MLD
MLD_VERSION	Enthält die Version der MLD, die installiert werden soll. Diese Version wird auch gegen die installierte Ubuntu-Version geprüft. Welche MLD-Version zu welcher Ubuntu-Version passt, ist in der Datei mld_ubuntu.dat gespeichert
myMLD_DIR	Verzeichnis, in dem dieses Programmpaket abgelegt ist. Die Variable ist in der Datei myMLD.cfg mit dem Verzeichnisnamen myMLD im Homeverzeichnis des Benutzers vorbelegt. Standardwert: ~/myMLD

[3] Zu finden unter http://www.minidvblinux.de, dann Download, Devel Version 3.0.3, SVN Repository und dort am Ende „all_sources"

Name	Beschreibung
PAKET_DIR	Verzeichnis, in dem die MLD-Entwicklungsumgebung die fertig gepackten Pakete ablegt. Dies ist abhängig von der verwendeten MLD-Version. Bislang habe ich noch keine Möglichkeit gefunden, das Verzeichnis automatisiert zu ermitteln. Deshalb muss der Wert entsprechend angepasst werden. Der Backslash im Standardwert dient zur Entwertung des folgenden Ausrufezeichens. Standardwert: $MLD_DIR/\!addons/3.0.3
SVN_DIR	Verzeichnis des gemeinsamen Ordners der virtuellen Maschine. Dort befindet sich bei mir die Workarea des KM-Tools (in meinem Fall Tortoise). Diese Variable wird vom Programm pack verwendet, um die Dateien bei Bedarf in das Verzeichnis zu kopieren, damit sie in Subversion eingecheckt werden können. Standardwert: /media/sf_SVN_WORKAREA
VDR1	Diese Variablen werden im Programm build verwendet, um die Menüeinträge für den Menüpunkt ISO erstellen zu generieren. Die erstellten ISOs werden auch mit dem Variableninhalt bezeichnet. **Wichtig** => Die Unterverzeichnisse, in dem die Dateien für den jeweiligen VDR liegen, müssen genauso benannt werden, wie der jeweilige hier vergebene Name. Sonst werden die Dateien vom Programm build nicht gefunden. Weiterhin muss dieser Name auch im jeweiligen Makefile.config für den Namen des zu erstellenden ISO vorhanden sein. Siehe auch Kapitel: eigene ISOs erstellen.
VDR2	
VDR3	
VDR4	
VDR5	

intern verwendete Variablen

Name	Beschreibung
ANFANG	Enthält einen Zeitstempel in Sekunden und merkt sich damit den Zeitpunkt, wann das Programm gestartet wurde.
AUFNAHME	Enthält die Benutzerauswahl aus dem Dialog, ob eine Aufnahme in das Config-Addon kopiert werden soll.
CODENAME_INST	Enthält den Namen der installierten Ubuntu-Distribution (z.B. oneiric, raring).
DAT_MLD_VERSION	Enthält die MLD_Version aus der aktuell verarbeiten Zeile der Datei mld_ubuntu.dat
DAT_UBUNTU_SOLL	Enthält die Ubuntu-Version aus der aktuell verarbeiten Zeile der Datei mld_ubuntu.dat, die zur MLD-Version passt
DAUER	Enthält die berechnete Laufzeit in Sekunden.
DEVELSERVER_AUSWAHL	Enthält die Benutzerauswahl, ob der Develserver gestartet oder beendet werden soll.
FN	Funktionsname => wird für das Logging verwendet, um den Bereich anzugeben, aus dem die Logmeldung generiert wurde. Hinweis: In FN darf kein Leerzeichen verwendet werden, weil die Funktion „log“ dies als mehrere Übergabewerte interpretiert. Abhilfe dafür habe ich noch nicht gefunden.
HARDWARE_INST	Enthält die Variante der installierten Ubuntu-Distribution (z.B. 32 oder 64 Bit).
HAUPT_AUSWAHL	Enthält die Benutzerauswahl aus dem Hauptmenü.
ISO_AUSWAHL	Enthält die Benutzerauswahl aus dem Dialog, welches ISO erstellt werden soll.
KOMPILIEREN	Enthält die Benutzerauswahl, ob die Entwicklungsumgebung nach dem Checkout auch gleich kompiliert werden soll.
MIN	Benötigte Laufzeit Minuten.
MLD_CFG	Name des aufzurufenden Programms, das die Anweisungen für die installierte Ubuntu Distribution enthält.
PASSWORT	Enthält das abgefragte Benutzerpasswort für Aktionen, in denen sudo-Rechte notwendig sind.
SEK	Benötigte Laufzeit Sekunden.
START_DIR	Enthält den Pfad, aus dem das Programm aufgerufen wurde, um beim Beenden wieder dorthin zu wechseln.
STD	Benötigte Laufzeit Stunden.
TAR_TEMP_DIR	Festgelegt ist dieses Verzeichnis auf ~/.tmp_unpack_$(date +%s) und dient zum Entpacken der TAR-Datei, die die MLD-Sourcen enthält. Das Verzeichnis wird vom Programm build bei Bedarf angelegt und auch wieder entfernt.
UBUNTU_VERSION_OK	Ein Merker, der TRUE enthält, wenn die Ubuntu-Version zur MLD-Version passt.
ZEIT	Zeitstempel der aktuellen Zeit in Sekunden. Wird benötigt, um die Laufzeit einzelner Programmteile berechnen zu können.
ZWISCHENZEIT	Zeitstempel der aktuellen Zeit in Sekunden. Wird benötigt, um die Laufzeit mehrerer Programmteile berechnen zu können (z.B. eines Hauptmenüpunktes).

Änderungshistorie

Datum	Version	Änderungen	Autor
13.07.2013	1.0	Neuerstellung	skippy
16.07.2013	1.1	MLD-Versionen eingeführt	skippy