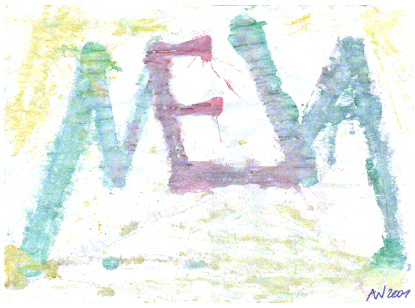
Albrecht Weinert

Linux Mint (14)

Installation on Workstations and Laptops for Development



Last mod.: 18.12.2012

Prof. Dr.-Ing. Albrecht Weinert a-weinert.de
weinert – automation weinert-automation.de

Labor für Medien und verteilte Anwendungen (MEVA-Lab) meva-lab.de
Fachbereich Informatik der Hochschule Bochum

Install Mint 14 on Workstations and Laptops for Development

V01.01, 01.11.2012:	new
V01.02, 06.12.2012:	first almost complete
V01.03, 10.12.2012:	more elaborate on Oracle Java
V01.04, 17.12.2012:	more elaborate on Mint 14 differences

Version: V1.04
Last modified by A. Weinert at 18.12.2012

Note on the template:	There is one common numbering for figures, lists, tables etc. (if present)
Note on version control:	SVN URL is https://ai2t.de/svn/albrecht/pub/Mint_for_Development.odt
Note on publications:	See also http://a-weinert.de/publication_en.html , http://blog.a-weinert.de/ and http://blog.a-weinert.de/allgemein/linux-mint-for-development/
This document's URL:	http://a-weinert.de/pub/Mint_for_Development.pdf. The version there might be newer if this is from elsewhere or on paper.

Table of content

1. Scope

This is about installing Linux Mint for development purposes on several platforms. As of this writing Mint 14 was the actual version – and used in the installations reported on.

1.1 Requirements

“For Development” in the title means to have

- SVN client also integrated in the actual explorer – whatever its actual name is in respective distribution – and integrated in Eclipse.

It means to have an actual

- Oracle JRE / JDK,
- AVR-tools also integrated in Eclipse,
- Network analysers (Wire shark etc.),
- Logic analysers (like Logic)

and much more.

For documentation compatible to other installations and platforms we want

- OpenOffice (not the enforced LibreOffice) and
- Windows compatible fonts.

And of course, not to have a

- browsers with the standard search engine,
- mail with calendar integration and multiple (imported) accounts and
- other every day usage tools

would be a shame.

In some cases the whole thing must be used and often also installed behind a

- non-transparent proxy and a
- company firewall which just opens the well known port for http(s) and mail.

Then FTP is open too, but may work in just one configuration not supported by all external FTP servers.

1.2 Platforms

Having many quit differing experiences with many distribution/ platform combinations this article reports on just Mint (12 ..14 mate, 64bit) on

- Fujitsu Siemens Lifebook E Series (2 Intel Core2, 2GHz, 2GB RAM, 125GB HDD),
- Fujitsu Lifebook S751 (4 Intel Core i7, 2.7GHz, 4GB RAM, 512GB HDD),
- Fujitsu Lifebook S782 (8 Intel Core i7, 2.1GHz, 16GB RAM, 256GB SSD only),
- Fujitsu Celsius W510 power (8 Intel Xeon, 3.4GHz, 8GB RAM, 500GB HDD).

Of course many of the tricks and experience reported can be transferred to other Linux distribution / platform cases, directly or with due modifications.

2. The summary – ex ante

For those not interested in the tips and details to follow (dynamically subjoined) but just in decision support I put the summary first. If you don't like the “wrong” place read on from the next chapter and come back here from the end.

Considering the differences in age, type and configuration of the three platforms reported here (and some others too) one sees hardly any varieties in installation and handling – except speed and audio / video quality. This is a big plus for Mint's flexibility, driver support and good hardware recognition. And the basic installation from the burned .iso DVD is strait and fast. The really hard part was always the location: behind a non-transparent proxy or not, and (almost impossible) dynamically change that in case of laptops.

Considering the distributions (12..14) 12 worked and installed well and 14 promises the same quality. 13 often meant trouble, mostly, on the same platforms. So I put it on the same list as 2000, ME, Vista … This seems a bit typical for Mint; the agile developers may neglect long term stability and robustness a bit too much for sake of their own flexibility and artistic freedom. Some symptoms point in that direction.

Mint documentation flamingly warns of upgrade installations (12 to 13/14 e.g.). One may tend (I did) to hardly believe this as other distributions do upgrade without troubles and partly automatically. Doing so with the standard recipe

```
apt-add-repository … sudo apt-get ….. upgrade …
```

ends almost reproducibly in an unusable system at the edge of total re-install or beyond.

Mint's notorious version incompatibility – Tip / Warning

That's one strong symptom for incompatibility between Mint versions. And besides leading to nowhere the upgrade installation takes ages. Instead of taking some (Python related) errors out of the standard text editor – gedit – you get a new one – pluma – looking totally the same. This point / this incompatibility may seem to minor to mention. But this renaming and moving around of tools and configuration files, often undocumented and without conceivable reason or advantage, breaks scripts, recipes and know how. This is good for any amount of frustration, loss of time and anger.

Another Mint weakness is the otherwise nice menu, changing its content – after removing / installing software with apt-.. (see requirements above) – is not really supported with different deficiencies in different releases, to put it mildly. The once proven repair –

```
apt-get alacarte
```

– destroyed the system by installation alone.

So beware of the instabilities and inconsistencies – take no proven tool or recipe as given after updates or upgrades. In that respect Microsoft Windows is not worse at all. That said Mint is a nice, comfortable distribution and a good candidate to replace Windows on Laptop and Workstations.

3. Installation

The basic installation is quite simple:

- Get the .iso image
  linuxmint-14.1-mate-dvd-64bit.iso or newer from a decent mirror,
- burn it,
- boot it.

Play a bit with it, especially

- set and check your keyboard layout
- check network connectivity (WLAN is sufficient),
- including proxy settings if necessary.

You should not go ahead with the machine if you can't type decently “/.,äöüÄÖÜß€<>|” or if you can't get to web with the pre-installed browser. Finally

- run the installer,
- give the appropriate answers leaving default whenever feasible.

This proceeding usually deletes all previous systems giving a clean fresh Mint.

As said above the upgrade installation is at least critical, and the normal installation deletes all target's previous data. Hence the data to be kept from the version to upgrade from – personal files, configuration data, applications, time consuming downloads etc. – have to be saved elsewhere and partly restored.

Fresh install with saved data – tip

If that's impractical one can use the system booted from the CD, to check and reduce the data to be saved. If its size would fit to a part of the target drive small enough, use the partition tool included (gparted) to do the following (may take some minutes):

- Shrink the partition used for the data to its minimum,
- move it to the end of (then) free space,
- make a formatted partition below / at the beginning for the new installation,
- move the boot attribute there,
  and give all partitions names.

Then let the installer not use all but the new made empty partition (installer may insist on formatting it again) and let it use the previous (kept) swap partition. This lets you mount and use the old data.

4. All behind a proxy

Some companies and universities let their users live behind a non-transparent proxy (the “non-transparent” being the problem hereby) and most companies, of course, keep their IT behind a company firewall. Often, even server and domain (AD) administrators / installators have no influence on those proxy's and firewall's settings – rendering all tips ODF sort “open port 11371“ totally useless.

If neither is the case with you go to next chapter. Otherwise this proxy & .. topic can get a real plague and even time-fuse bomb for Windows and Linux installations.

Seemingly friendly, Mint offers a central “proxy settings” in system menus – but this system setting is virtually useless as not respected by the majority of applications including some of Mint's core functions. This may be due to application's ignorance of the topic and / or the respective configuration entries changing names and places. A functional profile concept with automatic changes according to (W)LAN in reach – like Android has – is total utopia under that auspices.

Mint update

It uses wget. So do

```
sudo gedit /etc/wgetrc
```

Find the part where proxy is mentioned, uncomment the prepared settings and change them accordingly:

```
http_proxy = 185.175.85.10:8080
https_proxy = 185.175.85.10:8080
ftp_proxy = 185.175.85.10:8080
use_proxy = on
```

Listing 1: Proxy settings for update / wget.

APT

System settings won't work here too. So do.

```
gksudo gedit /etc/apt/apt.conf
```

and put there:

```
Acquire::http::proxy "http://193.175.85.10:8080/";
Acquire::https::proxy "https://193.175.85.10:8080/";
Acquire::ftp::proxy "ftp://193.175.85.10:8080/";
Acquire::socks::proxy "socks://193.175.85.10:8080/";
```

Listing 2: Proxy settings for apt.

SVN behind a proxy

Shamefully Subversion (SVN) doesn't respect system proxy settings of any platform – not even on those with a good automatic profile concept (as Andriod 4 e.g.). Hence, do

```
gedit /home/<user>/.subversion/servers
```

Put there something like

```
http-proxy-host = 185.175.85.10
http-proxy-port = 8080
```

LibreOffice or OpenOffice

Go to
Menu Tools (or Extras) options internet proxy
and do the appropriate settings there.

Firefox browser

On some distributions the default setting is “use system settings” is by itself sensible. If is found in preferences → advanced → network.

In quite seldom cases it may work for the browser. But in most cases (including 14.x pre-installed Firefox) it won't. And the only working variant will often be to put all settings “manual”.

To have internal servers work you might have to put all server names in the exception list, as IP addresses / ranges often proved insufficient.

Mint itself and proxies

And of course: Do not forget to set Mint's central proxy settings also. However useless they are, their still required.

To sum up:
The proxy topic in Mint is a total swamp. It makes a PC – a laptop ! – configured for “behind proxy” virtually useless in the “outside world” and vice versa.

A (bad) quantum of solace:
Windows' not much better here.

5. Java

Having the right JDK is fundamental for any development work in any way related to Java. Besides that many tools require or build on Java (a JRE) and it is often better to have one's own preference prepared and let them use it instead of letting many application installers each bring in their own Java.

First Java problem “Which”

Since Sun's decline one usually has and or needs two Java installations under Linux (or often has the wrong one).

```
<user>~ > java -version
java version "1.6.0_23"
OpenJDK Runtime Environment (IcedTea6 1.11pre)
(6b23~pre11-0ubuntu1.11.10.2)
OpenJDK 64-Bit Server VM (build 20.0-b11, mixed mode)
<user~ >
```

Listing 3: Command to check which java

Usually – as in the listing 3 – you'll find the OpenJDK variant installed. And it can be started directly by pure program's name from the command line.

Other variants – especially the one by Oracle – may be installed too and / or may be needed by some Java based applications. Which Javas are installed can most often be explored by searching for the tool jar:

```
<user>~ > find /usr/ -name jar
/usr/lib/jvm/java-6-openjdk/bin/jar
/usr/share/binfmts/jar
/usr/share/doc/openjdk-6-jre-headless/api/java/util/jar
<user>~ >
```

The exemplary three line outcome reveals only one variant installed. Here the second is a link to /etc/alternatives/jexec-binfmt which itself is a link to /etc/alternatives/jexec-binfmt which is a link to /usr/lib/jvm/java-6-openjdk/jre/lib/jar.binfmt which is magic text leading to /usr/bin/jexec which is a link to … we give it up and need an extra sub-chapter. (The first and third are „real“ jar tools)

Second problem “Where”

Linux has, like Windows, a concept of (search) path for executables (shell scripts and programmes). Windows has like Unix / Linux a concept of links. But, happily, they are used so seldom (and hidden so well), that many guys considering them self “Windows experts” would deny their existence. On the other hand, Linux uses links (alternatives) to a nightmarish extend often mingled inconsistently with its path “concept”.

When, in the listing above, we successfully ordered

```
<user>~ > java -version
```

we hit

```
/usr/bin/java
```

which is found on the path ($PATH by usr/bin's membership).

But this is not the programme, its a link to

```
/etc/alternatives/java
```

And even this is not not yet the programme, its a link to

```
/usr/lib/jvm/java-6-openjdk/jre/bin/java
```

And that is the progamme – i.e. the equivalent of

```
C:\util\jdk\bin\java.exe
```

in Windows, where

```
C:\util\jdk\bin\
```

would be on the path in the Windows example.

In your installation the exploration of “where is my Java” might even be more complex and yield one or two indirection stages more.

Remark by the way:	Those link chains can't make any system faster, can't they. And they can't make a mechanical disc drive live longer, probably.

Of course, one can (should!?) accept the linkage / alternative complexity as the (different) philosophy of the Linux world. But, no matter what philosophy, it should be implemented working.

If you enter jarsigner or jar (just to name two tools used in every decent Java project and installed with every decent JDK) you get an error.
The good news is: they are installed, as above find example showed.
The bad news: these two and some others have no (alternative) links. Ways out:

1. add them all to the linking (nightmare)
2. add /usr/lib/jvm/java-6-openjdk/jre/bin/ or the other to the path
   PATH=$PATH:/usr/lib/jvm/java-6-openjdk/jre/bin *)
   In that case one should delete all said links as redundant paths, eating resources and tending to mask otherwise obvious (path) errors. (Java Windows installations commit two similar sins which did often lead to endless trouble shooting.)
3. use always the full path like
   /usr/share/doc/openjdk-6-jre-headless/api/java/util/jar
   (No – not really!)

*) For making it permanent add this line to ~/.bashrc

After knowing which and where one can go ahead installing the necessary extensions, like Frame4J (ai2t.de/public/frame4j/) by:

```
<user>~> cd /usr/lib/jvm/java-1.6.0-openjdk
/usr/lib/jvm/java-1.6.0-openjdk > sudo ./bin/jar  xfv
/home/mapa/Downloads/erg.zip
```

But before you do install Java extensions or let Java be used by further tools yet to be installed you might want to change the Java installation.

Going to Oracle Java only

We saw as default Mint comes with OpenJDK. But for good – development, tooling etc. – reasons some prefer OracleJDK, and to have it alone. To get to this best follow instructions given by http://www.wikihow.com/Install-Oracle-Java-on-Ubuntu-Linux

You should always go there to see the actual version and variants fitting your requirements. If you do and follow the instructions there step by step you get what you want safely. On the other hand, you get this by combining both path and the link (alternate) approach, which is – at least – partly redundant.

Here is a shortened version of the above link's recipe for the 64 bit version:

Download the fitting JRE / JDK (the .tar.gz !) and the documentation (a .zip). Do

```
sudo apt-get purge openjdk-\*
```

and try to avoid the installation of any proposed substitute.

```
sudo mkdir -p /usr/local/java
cd /home/<user>/Downloads
sudo -s cp -r jdk-7u9-linux-x64.tar.gz /usr/local/java
sudo -s cp -r jre-7u9-linux-x64.tar.gz /usr/local/java
cd /usr/local/java
sudo -s chmod  a+x jdk-7u9-linux-x64.tar.gz
sudo -s chmod  a+x jre-7u9-linux-x64.tar.gz
sudo -s tar  xvzf  jdk-7u9-linux-x64.tar.gz
sudo -s tar  xvzf  jre-7u9-linux-x64.tar.gz

sudo gedit /etc/profile
```

and put the following text at the end:

```
JAVA_HOME=/usr/local/java/jdk1.7.0_09
JRE_HOME=/usr/local/java/jre1.7.0_09
PATH=$JAVA_HOME/bin:$PATH:$HOME/bin:$JRE_HOME/bin
export JAVA_HOME
export JRE_HOME
export PATH
```

Listing 4: End of /etc/profile for Oracle Java

The complete recipe for Oracle Java ends with the setting of the so called alternatives That's another name for Linux' linking nightmare. So you might as well decide to omit that (Listing 5) in the light of the already prepared PATH setting (which I always do).

```
    sudo update-alternatives --install "/usr/bin/java" "java"
"/usr/local/java/jre1.7.0_09/bin/java" 1
    sudo update-alternatives --install "/usr/bin/javac" "javac"
"/usr/local/java/jdk1.7.0_09/bin/javac" 1
    sudo update-alternatives --install "/usr/bin/javaws" "javaws"
"/usr/local/java/jre1.7.0_09/bin/javaws" 1
    sudo update-alternatives --set java /usr/local/java/jre1.7.0_09/bin/java
    sudo update-alternatives --set javac /usr/local/java/jdk1.7.0_09/bin/javac
    sudo update-alternatives --set javaws
/usr/local/java/jre1.7.0_09/bin/javaws
```

Listing 5: The linking / alternate part of the Oracle Java recipe
can / should be omitted when having a good $PATH setting.

Anyway (try to) run

```
. /etc/profile
```

or re-boot to get the above PATH (Listing 4) take effect.

Hint: If one takes the “path without linkage” approach suggested here usually requires all old Java alternate linkage remains to be killed. And it's usually better then, to use version neutral directory names, i.e.

```
JAVA_HOME=/usr/local/java/jdk
```

instead of

```
JAVA_HOME=/usr/local/java/jdk1.7.0_09
```

6. Subversion

Best install a fitting Collabnet Subversion (client) and care for the proxy (see above) if necessary. You may also just do

```
sudo apt-get install subversion
```

The bit of a Tortoise for the explorer is a bit more complicated, also due to the fact that Mint's explorer changes names and implementation every third version. Tortoise's Mint name is RabbitVCS and it should integrate to Nautilus (that is the Explorer) like Tortoise on Windows' Explorer. To get it and integrate might be hard and even dangerous as Mint's explorer names and implementations often arbitrarily change.
What worked in the end was:

```
sudo add-apt-repository ppa:rabbitvcs/ppa
sudo apt-get update
sudo apt-get install     rabbitvcs-nautilus3
sudo apt-get install     rabbitvcs-thunar
sudo apt-get install     rabbitvcs-gedit     (not necesary)
sudo apt-get install     rabbitvcs-cli       (not necesary)
sudo apt-get autoremove
```

And of course: restart.

For Mint 14 we have to do a little bit more :

```
cd /opt
su -
add-apt-repository ppa:rabbitvcs/ppa
apt-get update
apt-get install  rabbitvcs-nautilus3  python-caja  rabbitvcs-core
wget http://www.excellent.co.id/wp-content/uploads/2012/10/rabbit-caja.tar.gz
tar -zxvf rabbit-caja.tar.gz
cp caja* /usr/lib/pymodules/python2.7/rabbitvcs/util/
mkdir -p /usr/lib/caja/extensions-2.0/python/
cp RabbitVCS.py /usr/lib/caja/extensions-2.0/python/
```

And of course: restart.

Remark: Mint's explorer changes names – Nautillus, Caja, .. – and obstacles to install extensions like others change shirts. That's the background of the recipe change at Mint 14. And of course such breaking changes often takes hours to find the problem and then the solution.
Great thank to the person, who instead of joining the „There is no problem, only a stupid user"-chorus posted above solution under ([http://vavai.net/2012/10/solved-rabbitvcs-subversion-menu-on-nautiluscaja-linuxmint-13-maya/])

Extra: Script to update a lot of repos in one run

```
#!/bin/bash
echo -e "\n Update SVN repositories \n "

cd ~/svn_work/
echo -e "\n processing repos in $PWD : "
ls
echo -e " "
for f in albrecht  indUcDoc  indUcPr  we4misc
do
 cd ~/svn_work/$f
 echo "Processing repository $f"
 svn update
done
cd ~/eclipCeWS/
echo -e "\n processing repos in $PWD : "
ls
echo -e " "
for f in indWuL_proto_01
do
 cd ~/eclipCeWS/$f
 echo "Processing repository $f"
 svn update
done

java AskAlert -nb -wm "ready" "title= SVN updates"
```

Listing 6: Auto update repositories (a script template)

7. Eclipse

Install Eclipse by standard means and care for the proxy (see above) if necessary. The SVN integration might be a (javaHL) problem.

Additionally Eclipse on Linux spoils encoding of multi platform projects (to UTF-8). If existing (multi-platform) projects have another encoding, like ISO8859-1, one should this as global workspace setting before touching any project or file.

8. OpenOffice

The first step to get OpenOffice instead of pre-installed LibreOffice is always to remove LibreOffice by

```
sudo apt-get remove --purge libreoffice-core
```

For Mint 14 you must explicitly remove libreOffice's Java integration and uno packages also using the software manager. Afterwards just follow the instructions by

http://www.fossapps.com/2011/12/01/how-to-install-openoffice-in-ubuntu-11-10/

Additionally go to the folder desktop-integration and re-run.

The standard way of remove LibreOffice and then using the packet manager to get and install OpenOffice would fail as Linux Mint 14 repositories provide fake OpenOffice entries which de facto prepare for LibreOffice.

Having the same fonts as with Windows

- We live in a multi-platform environment.
- We want to co-operate with partners having mostly Windows and last but not least
- we like to be able to use own documents on our Windows machines also.

Linux (Mint) does not have the standard fonts, like Arial, CourrierNew and so on. If one of above points is true that deficiency soon gets very annoying.

```
sudo apt-get install msttcorefonts
```

or the usage of Mint's

```
software administration → fonts.
```

will get them. (Before 14 some extra work on configuration files might have been necessary.)

Get these standard fonts before touching any document or slides using those standard fonts. Or you might be in for disagreeable layout effects and big troubles with the other stakeholders of the respective document files.

9. Further settings

As lamented on above the Mint developers like to bring in discontinuities like instead of improving gedit bringing in a new look alike named pluma. Though being a trifle it breaks existing recipies and scripts – and a good load of those trifles add up to a annoyance. Instead of sudo apt-get install gedit (which gets gedit with its bugs) do:

```
sudo ln /usr/bin/pluma /usr/bin/gedit
```

10. Resume

See chapter 2 on page 4.

A Abbreviations

ACL access control list (List of rights on an object)

AD Active Directory (Microsoft's interpretation of LDAP)

AJAX Asynchronous JavaScript + XML

API Application Programme Interface

C/S Client-Server

CA Certification authority

CVS Concurrent Versioning System

FAQ Frequently Asked Questions

GSS Generic Security Service

GUID Globally Unique Identifier

GWT Google Webtoolkit, AJAX with Java only

HTML Hypertext Markup Language [RFC 1866]

HTTP Hypertext Transfer Protokoll

HTTPS HTTP via SSL

HW Hardware

IIOP Internet Inter-ORB Protocol

IP Internet Protocol

J2EE Java 2 Enterprise Edition

J2ME Java 2 Micro Edition

J2SE Java 2 Standard Edition

JAAS Java Authentication and Authorization Service

JAR Java Archive. (.zip + semantics)

JAXP Java API for XML Parsing

JCA Java Cryptography Architecture (Java Security API)

JCE Java Cryptography Extensions

JDBC Java Database Connectivity

JDK Java Development Kit

JEB Enterprise JavaBeans

JMX Java Management Extensions

JNDI	Java Naming and Directory services Interface
JNI	Java Native Interface
JRE	Java Runtime Environment; a JDK subset without development tools.
JSDK	Java Servlet Development Kit
JSF	Java Server Faces
JSP	Java Server Pages
JVM	Java virtual machine.
LAN	Local area network
LDAP	Lightweight Directory Access Protocol
LGPL	Lesser GNU Public License
MBean	Managed Bean (JMX)
MEVA	Labor für Medien und verteilte Anwendungen
MS	Microsoft
OMG	Object Management Group
OS	Operating System
PAM	Pluggable Authentication Module
PC	Personal Computer
R&D	Research and Development
RAID	Redundant Array of inexpensive Disks
RDF	Resource Description Framework (W3C)
RMI	Remote Method Invocation
RPC	Remote Procedure Call
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SQL	Structured query language, Datenbankbearbeitungssprache
SSL	Secure Socket Layer. Protokollschicht zu Absicherung.
SSO	Single Sign on; Authentifizierung vieler (n) Anwendungen gegen eine (1) "security realm".
SSPI	Security Support Provider Interface
SVN	Subversion
TCP	Transmission Control Protocol
TM	Trade Mark (Warenzeichen)

UML Unified Modelling Language

URI Uniform Resource Locator

WXP operating system Windows XP (MS)

W2K3 operating system Windows Server 2003 (MS)

W2K7 operating system Windows Server 2007 (MS)

W3 short for WWW

W3C World Wide Web Consortium

WebDAV Web-based Distributed Authoring and Versioning

WS Workstation

WSDL Web Services Description Language

XML eXtensible Markup Language