NAME

dbus-launch - Utility to start a message bus from a shell script

SYNOPSIS

dbus-launch [--version] [--help] [--sh-syntax] [--csh-syntax] [--auto-syntax] [--binary-syntax] [--close-stderr] [--exit-with-session] [--autolaunch=*MACHINEID*] [--config-file=*FILENAME*] [*PROGRAM*] [*ARGS...*]

DESCRIPTION

The **dbus-launch** command is used to start a session bus instance of *dbus-daemon* from a shell script. It would normally be called from a users login scripts. Unlike the daemon itself, **dbus-launch** exits, so backticks or the $() construct can be used to read information from **dbus-launch**.

With no arguments, **dbus-launch** will launch a session bus instance and print the address and PID of that instance to standard output.

You may specify a program to be run; in this case, **dbus-launch** will launch a session bus instance, set the appropriate environment variables so the specified program can find the bus, and then execute the specified program, with the specified arguments. See below for examples.

If you launch a program, **dbus-launch** will not print the information about the new bus to standard output.

When **dbus-launch** prints bus information to standard output, by default it is in a simple key-value pairs format. However, you may request several alternate syntaxes using the --sh-syntax, --csh-syntax, --binary-syntax, or --auto-syntax options. Several of these cause **dbus-launch** to emit shell code to set up the environment.

With the --auto-syntax option, **dbus-launch** looks at the value of the SHELL environment variable to determine which shell syntax should be used. If SHELL ends in csh, then csh-compatible code is emitted; otherwise Bourne shell code is emitted. Instead of passing --auto-syntax, you may explicitly specify a particular one by using --sh-syntax for Bourne syntax, or --csh-syntax for csh syntax. In scripts, its more robust to avoid --auto-syntax and you hopefully know which shell your script is written in.

See **http://www.freedesktop.org/software/dbus/** for more information about D-Bus. See also the man page for *dbus-daemon*.

EXAMPLES

Distributions running **dbus-launch** as part of a standard X session should run **dbus-launch --exit-with-session** after the X server has started and become available, as a wrapper around the main X client (typically a session manager or window manager), as in these examples:

dbus-launch --exit-with-session gnome-session

dbus-launch --exit-with-session openbox

dbus-launch --exit-with-session ~/.xsession

If your distribution does not do this, you can achieve similar results by running your session or window manager in the same way in a script run by your X session, such as ~/.xsession, ~/.xinitrc or ~/.Xclients.

To start a D-Bus session within a text(hymode session, do not use **dbus-launch**. Instead, see **dbus-run-session(1)**.

```
## test for an existing bus daemon, just to be safe
if test -z $DBUS_SESSION_BUS_ADDRESS ; then
## if not found, launch a new one
eval `dbus-launch --sh-syntax`
echo D-Bus per-session daemon address is: $DBUS_SESSION_BUS_ADDRESS
fi
```

Note that in this case, dbus-launch will exit, and dbus-daemon will not be terminated automatically on logout.

AUTOMATIC LAUNCHING

If DBUS_SESSION_BUS_ADDRESS is not set for a process that tries to use D-Bus, by default the process will attempt to invoke dbus-launch with the --autolaunch option to start up a new session bus or find the existing bus address on the X display or in a file in ~/.dbus/session-bus/

Whenever an autolaunch occurs, the application that had to start a new bus will be in its own little world; it can effectively end up starting a whole new session if it tries to use a lot of bus services. This can be suboptimal or even totally broken, depending on the app and what it tries to do.

There are two common reasons for autolaunch. One is ssh to a remote machine. The ideal fix for that would be forwarding of DBUS_SESSION_BUS_ADDRESS in the same way that DISPLAY is forwarded. In the meantime, you can edit the session.conf config file to have your session bus listen on TCP, and manually set DBUS_SESSION_BUS_ADDRESS, if you like.

The second common reason for autolaunch is an su to another user, and display of X applications running as the second user on the display belonging to the first user. Perhaps the ideal fix in this case would be to allow the second user to connect to the session bus of the first user, just as they can connect to the first users display. However, a mechanism for that has not been coded.

You can always avoid autolaunch by manually setting DBUS_SESSION_BUS_ADDRESS. Autolaunch happens because the default address if none is set is autolaunch:, so if any other address is set there will be no autolaunch. You can however include autolaunch in an explicit session bus address as a fallback, for example DBUS_SESSION_BUS_ADDRESS=something:,autolaunch: - in that case if the first address doesnt work, processes will autolaunch. (The bus address variable contains a comma-separated list of addresses to try.)

The --autolaunch option is considered an internal implementation detail of libdbus, and in fact there are plans to change it. Theres no real reason to use it outside of the libdbus implementation anyhow.

OPTIONS

The following options are supported:

--auto-syntax
Choose --csh-syntax or --sh-syntax based on the SHELL environment variable.

--binary-syntax
Write to stdout a nul-terminated bus address, then the bus PID as a binary integer of size sizeof(pid_t), then the bus X window ID as a binary integer of size sizeof(long). Integers are in the machines byte order, not network byte order or any other canonical byte order.

--close-stderr
Close the standard error output stream before starting the D-Bus daemon. This is useful if you want to capture dbus-launch error messages but you dont want dbus-daemon to keep the stream open to your application.

--config-file=FILENAME
Pass --config-file=FILENAME to the bus daemon, instead of passing it the --session argument. See the man page for dbus-daemon

--csh-syntax
Emit csh compatible code to set up environment variables.

--exit-with-session
If this option is provided, a persistent babysitter process will be created that watches stdin for HUP and tries to connect to the X server. If this process gets a HUP on stdin or loses its X connection, it kills the message bus daemon.

--autolaunch=MACHINEID
This option implies that **dbus-launch** should scan for a previously-started session and reuse the values found there. If no session is found, it will start a new session. The --exit-with-session option is implied if --autolaunch is given. This option is for the exclusive use of libdbus, you do not want to use it manually. It may change in the future.

--sh-syntax
Emit Bourne-shell compatible code to set up environment variables.

--version
Print the version of dbus-launch

--help
Print the help info of dbus-launch

NOTES

If you run **dbus-launch myapp** (with any other options), dbus-daemon will *not* exit when **myapp** terminates: this is because **myapp** is assumed to be part of a larger session, rather than a session in its own right.

AUTHOR

See **http://www.freedesktop.org/software/dbus/doc/AUTHORS**

BUGS

Please send bug reports to the D-Bus mailing list or bug tracker, see **http://www.freedesktop.org/software/dbus/**