```
**********************************************************
    7950 Thu Aug 14 14:05:27 2014
new/usr/src/tools/env/developer.sh
5092 env files don't need to define LOCKNAME by default
5091 illumos.sh env file's LOCKNAME definition is busted
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
#

#       Configuration variables for the runtime environment of the nightly
# build script and other tools for construction and packaging of releases.
# This script is sourced by 'nightly' and 'bldenv' to set up the environment
# for the build. This example is suitable for building a developers workspace,
# which will contain the resulting packages and archives. It is based off
# the onnv release. It sets NIGHTLY_OPTIONS to make nightly do:
#       check ELF ABI/versioning (-A)
#       runs 'make check' (-C)
#       DEBUG and non-DEBUG builds (-D)
#       runs lint in usr/src (-l plus the LINTDIRS variable)
#       sends mail on completion (-m and the MAILTO variable)
#       creates packages for PIT/RE (-p)
#       checks for changes in ELF runpaths (-r)
#
NIGHTLY_OPTIONS="-ACDlmpr";             export NIGHTLY_OPTIONS

# This is a variable for the rest of the script - GATE doesn't matter to
# nightly itself
GATE=onnv-bugfixes;                     export GATE

# CODEMGR_WS - where is your workspace at (or what should nightly name it)
CODEMGR_WS="/builds/$GATE";                     export CODEMGR_WS

# PARENT_WS is used to determine the parent of this workspace. This is
# for the options that deal with the parent workspace (such as where the
# proto area will go).
#
# If you use this, it must be local (or nfs): nightly cannot copy
# over ssh or http.
PARENT_WS="/ws/onnv-gate";                      export PARENT_WS

# CLONE_WS is the workspace nightly should do a bringover from.
CLONE_WS="ssh://anonhg@onnv.sfbay.sun.com//export/onnv-clone";  export CLONE_WS

# CLOSED_CLONE_WS is the workspace from which nightly should acquire
```

```
# the usr/closed tree.
CLOSED_CLONE_WS="${CLONE_WS}/usr/closed";       export CLOSED_CLONE_WS

# The bringover, if any, is done as STAFFER.
# Set STAFFER to your own login as gatekeeper or developer
# The point is to use group "staff" and avoid referencing the parent
# workspace as root.
# Some scripts optionally send mail messages to MAILTO.
#
STAFFER=nobody;                         export STAFFER
MAILTO=$STAFFER;                        export MAILTO

# The project (see project(4)) under which to run this build.  If not
# specified, the build is simply run in a new task in the current project.
BUILD_PROJECT=;                         export BUILD_PROJECT

# You should not need to change the next three lines
# You should not need to change the next four lines
LOCKNAME="'basename $CODEMGR_WS'_nightly.lock"; export LOCKNAME
ATLOG="$CODEMGR_WS/log";                        export ATLOG
LOGFILE="$ATLOG/nightly.log";                   export LOGFILE
MACH='uname -p';                                export MACH

# When the -A flag is specified, and ELF_DATA_BASELINE_DIR is defined,
# the ELF interface description file resulting from the build is compared
# to that from the specified directory. This ensures that our object
# versioning evolves in a backward compatible manner.
#
# You should not need to change this unless you wish to use locally cached
# baseline files. If you use this, it must be local (or nfs): nightly cannot
# copy over ssh or http.
#
ELF_DATA_BASELINE_DIR="/ws/onnv-gate/usr/src/ELF-data-baseline.$MACH";  export E

# This is usually just needed if the closed tree is missing, or when
# building a project gate with the -O (cap oh) flag.
# ON_CRYPTO_BINS="$PARENT_WS/packages/$MACH/on-crypto.$MACH.tar.bz2"
# export ON_CRYPTO_BINS

# REF_PROTO_LIST - for comparing the list of stuff in your proto area
# with. Generally this should be left alone, since you want to see differences
# from your parent (the gate).
#
REF_PROTO_LIST=$PARENT_WS/usr/src/proto_list_${MACH}; export REF_PROTO_LIST

#
#       build environment variables, including version info for mcs, motd,
# motd, uname and boot messages. Mostly you shouldn't change this except
# when the release slips (nah) or you move an environment file to a new
# release
#
ROOT="$CODEMGR_WS/proto/root_${MACH}";  export ROOT
SRC="$CODEMGR_WS/usr/src";              export SRC
VERSION="$GATE";                        export VERSION

#
# the RELEASE and RELEASE_DATE variables are set in Makefile.master;
# there might be special reasons to override them here, but that
# should not be the case in general
#
# RELEASE="5.10.1";                     export RELEASE
# RELEASE_DATE="October 2007";          export RELEASE_DATE

# proto area in parent for optionally depositing a copy of headers and
# libraries corresponding to the protolibs target
# not applicable given the NIGHTLY_OPTIONS
```

```
#
PARENT_ROOT=$PARENT_WS/proto/root_$MACH; export PARENT_ROOT
PARENT_TOOLS_ROOT=$PARENT_WS/usr/src/tools/proto/root_$MACH-nd; export PARENT_TO

#
# Package creation variables.  You probably shouldn’t change these,
# either.
#
# PKGARCHIVE determines where repositories will be created.
#
# PKGPUBLISHER* control the publisher settings for those repositories.
#
PKGARCHIVE="${CODEMGR_WS}/packages/${MACH}/nightly";    export PKGARCHIVE
# PKGPUBLISHER_REDIST="on-redist";                      export PKGPUBLISHER_REDI
# PKGPUBLISHER_NONREDIST="on-extra";                    export PKGPUBLISHER_NONR

# we want make to do as much as it can, just in case there’s more than
# one problem.
MAKEFLAGS=k;    export MAKEFLAGS

# Magic variable to prevent the devpro compilers/teamware from sending
# mail back to devpro on every use.
UT_NO_USAGE_TRACKING="1"; export UT_NO_USAGE_TRACKING

# Build tools - don’t set these unless you know what you’re doing.  These
# variables allows you to get the compilers and onbld files locally or
# through cachefs.  Set BUILD_TOOLS to pull everything from one location.
# Alternately, you can set ONBLD_TOOLS to where you keep the contents of
# SUNWonbld and SPRO_ROOT to where you keep the compilers.
#
#BUILD_TOOLS=/opt;                              export BUILD_TOOLS
#ONBLD_TOOLS=/opt/onbld;                        export ONBLD_TOOLS
#SPRO_ROOT=/opt/SUNWspro;                       export SPRO_ROOT

# This goes along with lint - it is a series of the form "A [y|n]" which
# means "go to directory A and run ’make lint’" Then mail me (y) the
# difference in the lint output. ’y’ should only be used if the area you’re
# linting is actually lint clean or you’ll get lots of mail.
# You shouldn’t need to change this though.
#LINTDIRS="$SRC y";     export LINTDIRS

#
# Reference to IA32 IHV workspace, proto area and packages
#
#IA32_IHV_WS=/ws/${GATE}-ihv;                           export IA32_IHV_WS
#IA32_IHV_ROOT=$IA32_IHV_WS/proto/root_i386;            export IA32_IHV_ROOT
#IA32_IHV_PKGS=$IA32_IHV_WS/packages/i386/nightly;      export IA32_IHV_PKGS

#
# Reference to binary-only IA32 IHV packages
#
#IA32_IHV_BINARY_PKGS=/ws/${GATE}-ihv-bin
#export IA32_IHV_BINARY_PKGS

# Set this flag to ’n’ to disable the automatic validation of the dmake
# version in use.  The default is to check it.
#CHECK_DMAKE=y

# Set this flag to ’n’ to disable the use of ’checkpaths’.  The default,
# if the ’N’ option is not specified, is to run this test.
#CHECK_PATHS=y

# Set this flag to ’y’ to enable the use of elfsigncmp to validate the
# output of elfsign.  Doing so requires that ’t’ be set in NIGHTLY_OPTIONS.
# The default is to not verify them.
#VERIFY_ELFSIGN=n
```

```
# BRINGOVER_FILES is the list of files nightly passes to bringover.
# If not set the default is "usr", but it can be used for bringing
# over deleted_files or other nifty directories.
#BRINGOVER_FILES="usr deleted_files"

# POST_NIGHTLY can be any command to be run at the end of nightly.  See
# nightly(1) for interactions between environment variables and this command.
#POST_NIGHTLY=
**********************************************************
    8563 Thu Aug 14 14:05:27 2014
new/usr/src/tools/env/gatekeeper.sh
5092 env files don’t need to define LOCKNAME by default
5091 illumos.sh env file’s LOCKNAME definition is busted
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
#

#       Configuration variables for the runtime environment of the nightly
# build script and other tools for construction and packaging of releases.
# This script is sourced by ’nightly’ and ’bldenv’ to set up the environment
# for the build. This example is suitable for building a gate,
# which will contain the resulting packages and archives (builds of the gate
# are done in children and then the resulting archives, packages, and proto
# area are put into the parent for everyone to use). It is based off
# the onnv release. It sets NIGHTLY_OPTIONS to make nightly do:
#       DEBUG and non-DEBUG builds (-D)
#       creates packages for PIT/RE (-p)
#       checks for new interfaces in libraries (-A)
#       runs ’make check’ (-C)
#       runs lint in usr/src (-l plus the LINTDIRS variable)
#       sends mail on completion (-m and the MAILTO variable)
#       updates the protolist in the parent for children to compare with (-u)
#       updates the proto area in the parent when done (-U)
#       checks for changes in ELF runpaths (-r)
#       checks for changes in unreferenced files (-f)
#
NIGHTLY_OPTIONS="-ADClmpuUrf";          export NIGHTLY_OPTIONS

# This is a variable for the rest of the script - GATE doesn’t matter to
# nightly itself
GATE=onnv-gate;                                 export GATE

# CODEMGR_WS - where is your workspace at (or what should nightly name it)
# there is only one definition here, which assumes all the gate build machines
# (sparc and x86) are set up the same. But remember, this is a script, so
# you _could_ look at $MACH or ‘uname -n‘ and set these variables differently.
CODEMGR_WS="/builds/$GATE";                     export CODEMGR_WS

# PARENT_WS is used to determine the parent of this workspace. This is
# for the options that deal with the parent workspace (such as where the
# proto area will go).
#
```

```
# If you use this, it must be local (or nfs): nightly cannot copy
# over ssh or http.
PARENT_WS="/ws/$GATE";                          export PARENT_WS

# CLONE_WS is the workspace nightly should do a bringover from.
CLONE_WS="ssh://anonhg@onnv.sfbay.sun.com//export/onnv-clone";  export CLONE_WS

# CLOSED_CLONE_WS is the workspace from which nightly will acquire the
# usr/closed tree.
CLOSED_CLONE_WS="${CLONE_WS}/usr/closed"
export CLOSED_CLONE_WS

# The bringover, if any, is done as STAFFER.
# Set STAFFER to your own login as gatekeeper or integration engineer.
# The point is to use group "staff" and avoid referencing the parent
# workspace as root.
# Some scripts optionally send mail messages to MAILTO.
#
STAFFER=nobody;                         export STAFFER
MAILTO=$STAFFER;                        export MAILTO

# The project (see project(4)) under which to run this build.  If not
# specified, the build is simply run in a new task in the current project.
BUILD_PROJECT=;                         export BUILD_PROJECT

# You should not need to change the next three lines
# You should not need to change the next four lines
LOCKNAME="‘basename $CODEMGR_WS‘_nightly.lock"; export LOCKNAME
ATLOG="$CODEMGR_WS/log";                        export ATLOG
LOGFILE="$ATLOG/nightly.log";                   export LOGFILE
MACH=‘uname -p‘;                                export MACH

# When the -A flag is specified, and ELF_DATA_BASELINE_DIR is defined,
# the ELF interface description file resulting from the build is compared
# to that from the specified directory. This ensures that our object
# versioning evolves in a backward compatible manner.
#
# You should not need to change this unless you wish to use locally cached
# baseline files. If you use this, it must be local (or nfs): nightly cannot
# copy over ssh or http.
#
ELF_DATA_BASELINE_DIR="/ws/onnv-gate/usr/src/ELF-data-baseline.$MACH";  export E

# This is usually just needed if the closed tree is missing, or when
# building a project gate with the -O (cap oh) flag.
# ON_CRYPTO_BINS="$PARENT_WS/packages/$MACH/on-crypto.$MACH.tar.bz2"
# export ON_CRYPTO_BINS

# REF_PROTO_LIST - for comparing the list of stuff in your proto area
# with. Generally this should be left alone, since you want to see differences
# between todays build and yesterdays.
#
REF_PROTO_LIST=$PARENT_WS/usr/src/proto_list_${MACH}; export REF_PROTO_LIST

#
#       build environment variables, including version info for mcs, motd,
# motd, uname and boot messages. Mostly you shouldn’t change this except
# when the release slips (nah) or when starting a new release.
#
ROOT="$CODEMGR_WS/proto/root_${MACH}";  export ROOT
SRC="$CODEMGR_WS/usr/src";              export SRC
VERSION="$GATE";                        export VERSION

#
# the RELEASE and RELEASE_DATE variables are set in Makefile.master;
# there might be special reasons to override them here, but that
# should not be the case in general
#
# RELEASE="5.10.1";                     export RELEASE
# RELEASE_DATE="October 2007";          export RELEASE_DATE

# proto area in parent for optionally depositing a copy of headers and
# libraries corresponding to the protolibs target
#
PARENT_ROOT=$PARENT_WS/proto/root_$MACH; export PARENT_ROOT
PARENT_TOOLS_ROOT=$PARENT_WS/usr/src/tools/proto/root_$MACH-nd; export PARENT_TO

#
# Package creation variables.  You probably shouldn't change these,
# either.
#
# PKGARCHIVE determines where repositories will be created.
#
# PKGPUBLISHER* control the publisher settings for those repositories.
#
PKGARCHIVE="${PARENT_WS}/packages/${MACH}/nightly";     export PKGARCHIVE
# PKGPUBLISHER_REDIST="on-nightly";                     export PKGPUBLISHER_REDI
# PKGPUBLISHER_NONREDIST="on-extra";                    export PKGPUBLISHER_NONR


# we want make to do as much as it can, just in case there's more than
# one problem. This is especially important with the gate, since multiple
# unrelated broken things can be integrated.
MAKEFLAGS=k;    export MAKEFLAGS

# Magic variable to prevent the devpro compilers/teamware from sending
# mail back to devpro on every use.
UT_NO_USAGE_TRACKING="1"; export UT_NO_USAGE_TRACKING

# Build tools - don't set these unless you know what you're doing.  These
# variables allows you to get the compilers and onbld files locally or
# through cachefs.  Set BUILD_TOOLS to pull everything from one location.
# Alternately, you can set ONBLD_TOOLS to where you keep the contents of
# SUNWonbld and SPRO_ROOT to where you keep the compilers.
#
#BUILD_TOOLS=/opt;                              export BUILD_TOOLS
#ONBLD_TOOLS=/opt/onbld;                        export ONBLD_TOOLS
#SPRO_ROOT=/opt/SUNspro;                        export SPRO_ROOT

# This goes along with lint - it is a series of the form "A [y|n]" which
# means "go to directory A and run 'make lint'" Then mail me (y) the
# difference in the lint output. 'y' should only be used if the area you're
# linting is actually lint clean or you'll get lots of mail.
# You shouldn't need to change this though.
#LINTDIRS="$SRC y";     export LINTDIRS

#
# Reference to IA32 IHV workspace, proto area and packages
#
#IA32_IHV_WS=/ws/${GATE}-ihv;                           export IA32_IHV_WS
#IA32_IHV_ROOT=$IA32_IHV_WS/proto/root_i386;            export IA32_IHV_ROOT
#IA32_IHV_PKGS=$IA32_IHV_WS/packages/i386/nightly;      export IA32_IHV_PKGS

#
# Reference to binary-only IA32 IHV packages
#
#IA32_IHV_BINARY_PKGS=/ws/${GATE}-ihv-bin
#export IA32_IHV_BINARY_PKGS

# Set this flag to 'n' to disable the automatic validation of the dmake
# version in use.  The default is to check it.
#CHECK_DMAKE=y
```

```

# Set this flag to 'n' to disable the use of 'checkpaths'.  The default,
# if the 'N' option is not specified, is to run this test.
#CHECK_PATHS=y

# Set this flag to 'y' to enable the use of elfsigncmp to validate the
# output of elfsign.  Doing so requires that 't' be set in NIGHTLY_OPTIONS.
# The default is to not verify them.
#VERIFY_ELFSIGN=n

# BRINGOVER_FILES is the list of files nightly passes to bringover.
# If not set the default is "usr", but it can be used for bringing
# over deleted_files or other nifty directories.
#BRINGOVER_FILES="usr deleted_files"

# POST_NIGHTLY can be any command to be run at the end of nightly.  See
# nightly(1) for interactions between environment variables and this command.
#POST_NIGHTLY=
```

****
8410 Thu Aug 14 14:05:27 2014
new/usr/src/tools/env/illumos.sh
5092 env files don’t need to define LOCKNAME by default
5091 illumos.sh env file’s LOCKNAME definition is busted
****
______unchanged_portion_omitted_

```
maxjobs DMAKE_MAX_JOBS # "DMAKE_MAX_JOBS" passed as ksh(1) name reference
export DMAKE_MAX_JOBS

# path to onbld tool binaries
ONBLD_BIN='/opt/onbld/bin'

# PARENT_WS is used to determine the parent of this workspace. This is
# for the options that deal with the parent workspace (such as where the
# proto area will go).
export PARENT_WS=''

# CLONE_WS is the workspace nightly should do a bringover from.
export CLONE_WS='ssh://anonhg@hg.illumos.org/illumos-gate'

# The bringover, if any, is done as STAFFER.
# Set STAFFER to your own login as gatekeeper or developer
# The point is to use group "staff" and avoid referencing the parent
# workspace as root.
# Some scripts optionally send mail messages to MAILTO.
#
export STAFFER="$LOGNAME"
export MAILTO="$STAFFER"

# If you wish the mail messages to be From: an arbitrary address, export
# MAILFROM.
#export MAILFROM="user@example.com"

# The project (see project(4)) under which to run this build.  If not
# specified, the build is simply run in a new task in the current project.
export BUILD_PROJECT=''

# You should not need to change the next three lines
# You should not need to change the next four lines
export LOCKNAME="$(basename -- "$CODEMGR_WS")_nightly.lock"
export ATLOG="$CODEMGR_WS/log"
export LOGFILE="$ATLOG/nightly.log"
export MACH="$(uname -p)"

#
#  The following two macros are the closed/crypto binaries.  Once
#  Illumos has totally freed itself, we can remove these references.
#
# Location of encumbered binaries.
export ON_CLOSED_BINS="$CODEMGR_WS/closed"
# Location of signed cryptographic binaries.
export ON_CRYPTO_BINS="$CODEMGR_WS/on-crypto.$MACH.tar.bz2"

# REF_PROTO_LIST - for comparing the list of stuff in your proto area
# with. Generally this should be left alone, since you want to see differences
# from your parent (the gate).
#
export REF_PROTO_LIST="$PARENT_WS/usr/src/proto_list_${MACH}"


export ROOT="$CODEMGR_WS/proto/root_${MACH}"
export SRC="$CODEMGR_WS/usr/src"
export MULTI_PROTO="no"
```

```
#
#       build environment variables, including version info for mcs, motd,
# motd, uname and boot messages. Mostly you shouldn't change this except
# when the release slips (nah) or you move an environment file to a new
# release
#
export VERSION="$GATE"

#
# the RELEASE and RELEASE_DATE variables are set in Makefile.master;
# there might be special reasons to override them here, but that
# should not be the case in general
#
# export RELEASE='5.11'
# export RELEASE_DATE='October 2007'

# proto area in parent for optionally depositing a copy of headers and
# libraries corresponding to the protolibs target
# not applicable given the NIGHTLY_OPTIONS
#
export PARENT_ROOT="$PARENT_WS/proto/root_$MACH"
export PARENT_TOOLS_ROOT="$PARENT_WS/usr/src/tools/proto/root_$MACH-nd"

# Package creation variables.  You probably shouldn't change these,
# either.
#
# PKGARCHIVE determines where the repository will be created.
#
# PKGPUBLISHER_REDIST controls the publisher setting for the repository.
#
export PKGARCHIVE="${CODEMGR_WS}/packages/${MACH}/nightly"
# export PKGPUBLISHER_REDIST='on-redist'

# Package manifest format version.
export PKGFMT_OUTPUT='v1'

# we want make to do as much as it can, just in case there's more than
# one problem.
export MAKEFLAGS='k'

# Magic variable to prevent the devpro compilers/teamware from sending
# mail back to devpro on every use.
export UT_NO_USAGE_TRACKING='1'

# Build tools - don't change these unless you know what you're doing.  These
# variables allows you to get the compilers and onbld files locally or
# through cachefs.  Set BUILD_TOOLS to pull everything from one location.
# Alternately, you can set ONBLD_TOOLS to where you keep the contents of
# SUNWonbld and SPRO_ROOT to where you keep the compilers.  SPRO_VROOT
# exists to make it easier to test new versions of the compiler.
export BUILD_TOOLS='/opt'
#export ONBLD_TOOLS='/opt/onbld'
export SPRO_ROOT='/opt/SUNWspro'
export SPRO_VROOT="$SPRO_ROOT"

# This goes along with lint - it is a series of the form "A [y|n]" which
# means "go to directory A and run 'make lint'" Then mail me (y) the
# difference in the lint output. 'y' should only be used if the area you're
# linting is actually lint clean or you'll get lots of mail.
# You shouldn't need to change this though.
#export LINTDIRS="$SRC y"

# Set this flag to 'n' to disable the automatic validation of the dmake
# version in use.  The default is to check it.
#CHECK_DMAKE='y'
```

```
# Set this flag to ’n’ to disable the use of ’checkpaths’.  The default,
# if the ’N’ option is not specified, is to run this test.
#CHECK_PATHS=’y’

# POST_NIGHTLY can be any command to be run at the end of nightly.  See
# nightly(1) for interactions between environment variables and this command.
#POST_NIGHTLY=

# Uncomment this to disable support for SMB printing.
# export ENABLE_SMB_PRINTING=’#’
```