Build jobs

Files

All files related to building should be places inside the related project source in the testing directory.

Build scripts file names should all use the template build-<project>.sh and be lower case.

If the project being built is an external source the build scripts for that project can be placed in the first project that depends on that source. If many projects depend on a project and they don't depend on each other so a chain of dependency can be established you can place that project outside of a related project.

Build examples / templates

Building external source

This example fetches are tar ball from the Internet, checks the consistency and builds.

```
#!/usr/bin/env bash
source `dirname "$0"`/lib.sh && init || exit 1

LDNS="ldns-1.6.12"
LDNS_URL="http://nlnetlabs.nl/downloads/ldns/$LDNS.tar.gz"
LDNS_FILENAME="$LDNS.tar.gz"
LDNS_HASH_TYPE="sha1"
LDNS_HASH="1d61df0f666908551d5a62768f77d63e727810aa"

check_if_built ldns && exit 0
start_build ldns

LDNS_SRC=`fetch_src "$LDNS_URL" "$LDNS_FILENAME" "$LDNS_HASH_TYPE" "$LDNS_HASH"`

build_ok=0
(
    gunzip -c "$LDNS_SRC" | tar xf - &&
    cd "$LDNS" &&
    ./configure --prefix="$INSTALL_ROOT" \
        --disable-gost &&
    $MAKE &&
    $MAKE install
) &&
build_ok=1
;;

if [ "$build_ok" -eq 1 ]; then
    set_build_ok ldns || exit 1
    exit 0
fi

exit 1
```

Building SVN source

This example is similar to building from external source just that the source has already been checked out in the WORKSPACE by Jenkins.

```
#!/usr/bin/env bash
source `dirname "$0"`/lib.sh && init || exit 1

check_if_built softhsm && exit 0
start_build softhsm

build_ok=0
(
    sh autogen.sh &&
    mkdir -p build &&
    cd build &&
    ../configure --prefix="$INSTALL_ROOT" &&
    $MAKE &&
    $MAKE check &&
    $MAKE install
) &&
build_ok=1

if [ "$build_ok" -eq 1 ]; then
    set_build_ok softhsm || exit 1
    exit 0
fi

exit 1
```

Building with dependency

To depend on other Jenkins projects / jobs that was built before this job you can use 'require', it will check that the required component was built successfully and it will add that components SVN_REVISION to this jobs SVN_REVISION making this build unique with regards to the required component.

```
#!/usr/bin/env bash
source `dirname "$0"`/lib.sh && init || exit 1

require ldns
require softhsm

check_if_built opendnssec && exit 0
start_build opendnssec

build_ok=0
...
```

Handling differences between distributions

Even if the framework tries to eliminate/handle most of the differences between the distributions you some times needs to specify different configure operations or add CFLAGS for just one distributions, this can be easily done with the help of DISTRIBUTION and a 'case'.

Handling prebuild options:

```
case "$DISTRIBUTION" in
    openbsd )
        export AUTOCONF_VERSION="2.68"
        export AUTOMAKE_VERSION="1.11"
        ;;
    sunos | \
    netbsd )
        append_cflags "-std=c99"
        ;;
    opensuse )
        append_ldflags "-lncurses"
        ;;
esac
```

Handling build options:

```
case "$DISTRIBUTION" in
    centos | \
    redhat | \
    fedora | \
    sl | \
    ubuntu | \
    debian | \
    opensuse | \
    freebsd | \
    sunos | \
    openbsd )
        (
            gunzip -c "$LDNS_SRC" | tar xf - &&
            cd "$LDNS" &&
            ./configure --prefix="$INSTALL_ROOT" \
                --disable-gost &&
            $MAKE &&
            $MAKE install
        ) &&
        build_ok=1
        ;;
    netbsd )
        (
            gunzip -c "$LDNS_SRC" | tar xf - &&
            cd "$LDNS" &&
            patch -p1 < ../ldns-1.6.12-doxyparse.pl-netbsd.patch &&
            ./configure --prefix="$INSTALL_ROOT" \
                --disable-gost &&
            $MAKE &&
            $MAKE install
        ) &&
        build_ok=1
        ;;
esac
```