Yocto cheatsheet

Related topics

- Simplify yocto project setup using kas (yocto-kas.html)
- Create bitbake recipes (creating-recipe.html)

conf/local.conf - frequent configurations

- Adding a package (IMAGE_INSTALL (https://docs.yoctoproject.org/ref-manual/variables.html#term-IMAGE_INSTALL)):

```
IMAGE_INSTALL:append = " wireguard-module rsyslog"    # syntax for "Honister" or
newer releases
IMAGE_INSTALL_append = " wireguard-module rsyslog"    # syntax for releases older
than "Honister"
```

- Configuring the hardware target (MACHINE (https://docs.yoctoproject.org/ref-manual/variables.html#term-MACHINE)):

```
MACHINE = "raspberrypi4-64"
```

- Adding a rootfs format (IMAGE_FSTYPES (https://docs.yoctoproject.org/ref-manual/variables.html#term-IMAGE_FSTYPES)):

```
IMAGE_FSTYPES:append = " hddimg"    # syntax for "Honister" or newer releases
IMAGE_FSTYPES_append = " hddimg"    # syntax for releases older than "Honister"
```

- Don't fail on dangling `*.bbappend` files (BB_DANGLINGAPPENDS_WARNONLY (https://docs.yoctoproject.org/ref-manual/variables.html#term-BB_DANGLINGAPPENDS_WARNONLY)):

```
BB_DANGLINGAPPENDS_WARNONLY = "true"
```

- Set useful features (EXTRA_IMAGE_FEATURES (https://docs.yoctoproject.org/ref-manual/variables.html#term-EXTRA_IMAGE_FEATURES)):

```
EXTRA_IMAGE_FEATURES = "debug-tweaks ssh-server-dropbear package-management"
```

- Switch the kernel recipe (PREFERRED_PROVIDER (https://docs.yoctoproject.org/ref-manual/variables.html#term-PREFERRED_PROVIDER), PREFERRED_VERSION (https://docs.yoctoproject.org/ref-manual/variables.html#term-PREFERRED_VERSION)):

```
PREFERRED_PROVIDER_virtual/kernel = "linux-raspberrypi-rt"
PREFERRED_VERSION_virtual/kernel = "4.19.%"
```

- RaspberryPi extra flags (ENABLE_I2C (https://meta-raspberrypi.readthedocs.io/en/latest/extra-build-config.html#enable-i2c),

RPI_EXTRA_CONFIG (https://meta-raspberrypi.readthedocs.io/en/latest/extra-build-config.html#manual-additions-to-config-txt)):

```
ENABLE_I2C = "1"
RPI_EXTRA_CONFIG:append = "\n \
dtoverlay=i2c-rtc,ds3231 \n \
"    # syntax for "Honister" or newer releases
RPI_EXTRA_CONFIG_append = "\n \
dtoverlay=i2c-rtc,ds3231 \n \
"    # syntax for releases older than "Honister"
```

- Appending to a variable when building for a specific architecture/hardware target:

```
IMAGE_INSTALL:append:qemuall = " ntpdate "    # syntax for "Honister" or newer releases
IMAGE_INSTALL_append_qemuall = " ntpdate "    # syntax for releases older than "Honister"
```

- Removing from a variable when building for a specific architecture/hardware target:

```
IMAGE_INSTALL:remove:qemuall = "avrdude"    # syntax for "Honister" or newer releases
IMAGE_INSTALL_remove_qemuall = "avrdude"    # syntax for releases older than "Honister"
```

- Setting the default hostname (reference (https://docs.yoctoproject.org/dev-manual/common-tasks.html#customizing-an-image-hostname)):

```
hostname:pn-base-files = "custom hostname"    # syntax for "Honister" or newer releases
hostname_pn-base-files = "custom hostname"    # syntax for releases older than "Honister"
```

- Ignoring some recipes (BBMASK (https://docs.yoctoproject.org/ref-manual/variables.html#term-BBMASK)):

```
BBMASK = "meta-freescale/recipes-kernel/linux"
BBMASK .= "|meta-freescale/recipes-connectivity/openssl"
```

- Blacklisting a package based on name (syntax from Kirkstone):

```
SKIP_RECIPE[busybox] = "Lorem ipsum error message"
```

- Blacklisting packages based on license (INCOMPATIBLE_LICENSE (https://docs.yoctoproject.org/ref-manual/variables.html#term-INCOMPATIBLE_LICENSE)):

```
INCOMPATIBLE_LICENSE = "GPL-3.0 LGPL-3.0 AGPL-3.0"
```

- Whitelisting packages with commercial license flags (LICENSE_FLAGS_WHITELIST (https://docs.yoctoproject.org/ref-manual/variables.html#term-LICENSE_FLAGS_WHITELIST)):

```
LICENSE_FLAGS_WHITELIST = "commercial_ffmpeg commercial_x264" # Honister or older
LICENSE_FLAGS_ACCEPTED = "commercial_ffmpeg commercial_x264" # Kirkstone or newer
```

conf/local.conf - mirroring & caching

- **Enabling the low-level.wiki mirrors**:

```
SOURCE_MIRROR_URL = "https://low-level.wiki/large_files/yocto/downloads/"
INHERIT += "own-mirrors"
```

- Setting a pre-mirror server (**SOURCE_MIRROR_URL (https://docs.yoctoproject.org/ref-manual/variables.html#term-SOURCE_MIRROR_URL)**):

```
SOURCE_MIRROR_URL = "http://.../sources/"
INHERIT += "own-mirrors"
```

- Setting a sstate-mirror server (do not change the `PATH` - it's a keyword, **SSTATE_MIRRORS (https://docs.yoctoproject.org/ref-manual/variables.html#term-SSTATE_MIRRORS)**):

```
SSTATE_MIRRORS = "file://.* http://.../sstate/PATH"
```

- Setting the local `DL_DIR` (this can be reused across multiple builds or to set up a mirror, **DL_DIR (https://docs.yoctoproject.org/ref-manual/variables.html#term-DL_DIR)**, **BB_GENERATE_MIRROR_TARBALLS (https://docs.yoctoproject.org/ref-manual/variables.html#term-BB_GENERATE_MIRROR_TARBALLS)**):

```
DL_DIR = "/var/data/downloads"
BB_GENERATE_MIRROR_TARBALLS = "1"
```

- Setting the local `SSTATE_DIR` (this can be reused across multiple builds or to set up a mirror, **SSTATE_DIR (https://docs.yoctoproject.org/ref-manual/variables.html#term-SSTATE_DIR)**):

```
SSTATE_DIR = "/var/data/sstate"
```

Other Yocto commands

- Search engine for yocto programs, recipes and layers: **https://layers.openembedded.org/layerindex/branch/master/recipes/ (https://layers.openembedded.org/layerindex/branch/master/recipes/)**.

- Run kernel menuconfig (you have to save and copy the generated config file afterwards):

```
bitbake -c menuconfig virtual/kernel
```

- Clean the environment for a recipe:

```
bitbake -c cleanall avrdude
```

- Download all the prerequisites for a build:

```
bitbake core-image-minimal --runall=fetch
```

- Find out which recipe is responsible for a file in the final image:

```
oe-pkgdata-util find-path /bin/sh
```

- Find out which recipe is responsible for a package:

```
oe-pkgdata-util lookup-recipe python3-io
```

- Find out information (recipe name, version) about a package:

```
oe-pkgdata-util package-info acl
```

- Find out the value of some variables from a recipe:

```
oe-pkgdata-util read-value RDEPENDS acl
```

- Start a docker container with the container's directory `/home` mapped over the physical directory `/mnt/ssd/rpi-tutorial-2/`. This is useful if you're recreating old yocto builds, and the host OS is newer (and incompatible). The re-mapping is specified, in case the default docker filesystem is placed on a HDD, but you also have an SSD installed.

```
sudo docker run -it --mount type=bind,source=/mnt/ssd/rpi-tutorial-2/,target=/home
ubuntu:16.04
```

Using generated images

- Run image in qemu (without root permissions or graphical interface, after the yocto environment has been "sourced", check the **README (http://git.yoctoproject.org/cgit.cgi/poky/tree/scripts/runqemu.README)**). Only works if the board has an acompanying `*.qemuboot.conf` file:

```
runqemu nographic slirp
```

- Copy the generated image for RaspberryPi on sdcard:

```
sudo apt-get install bmap-tools
sudo bmaptool copy core-image-minimal-raspberrypi4-64.wic.bz2 /dev/mmcblk0
```

- Converting from "hddimg" image to "vdi" (loadable into virtualbox):

```
"C:\Program Files\Oracle\VirtualBox\VBoxManage.exe" convertdd core-image-
minimal.hddimg  new_file.vdi --format VDI
```

- Converting from "vmdk" image to raw "hddimg" (writable on a disk):

```
vboxmanage clonemedium disk ubuntu-cosmic-18.10-cloudimg.vmdk --format RAW pm-vm-
physical-1.0.img
```

Using "templateconf"

- Create a new folder.
- Use the **poky/meta-poky/conf (http://git.yoctoproject.org/cgit/cgit.cgi/poky/tree/meta-poky/conf)** or **poky/meta-debian/conf (https://github.com/meta-debian/meta-debian/tree/master/conf)** folders as references. You only need `bblayers.conf.sample` and `local.conf.sample`. The `bblayers.conf.sample` file uses `##OEROOT##` placeholder to point to the `poky` folder. The `conf-notes.txt` file is optional.
- Use it (`TEMPLATECONF` must point to the configuration directory):

```
TEMPLATECONF=meta-.../.../conf source ./oe-init-build-env
```