```
#!/bin/bash
# shellcheck source=kz_common.sh
##############################################################################
# Build development environment.
#
# Written by Karel Zimmer <info@karelzimmer.nl>, CC0 1.0 Universal
# <https://creativecommons.org/publicdomain/zero/1.0>, 2023.
##############################################################################

##############################################################################
# Import
##############################################################################

program_path=$(cd "$(dirname "$(realpath "$0")")" && pwd)
source "$program_path"/kz_common.sh 2> >(systemd-cat) || exit 1


##############################################################################
# Variables
##############################################################################

program_name='kz-getdev'
program_desc=$(gettext 'Build development environment')
display_name=${program_name/kz-/kz }

usage=$(eval_gettext "Usage: \$display_name \$options_usage")
help="$(eval_gettext "Usage: \$display_name [OPTION...]")

$program_desc.

$(gettext 'Options:')
$options_help"


##############################################################################
# Functions
##############################################################################

function check_input {
    local -i rc=0
    local parsed=''

    parsed=$(
        getopt  --alternative                       \
                --options       "$options_short"    \
                --longoptions   "$options_long"     \
                --name          "$display_name"     \
                --              "$@"
        ) || rc=$?
    if [[ $rc -ne $ok ]]; then
        printf '%s\n' "$usage_line"
        exit $error
    fi
    eval set -- "$parsed"
    process_option "$@"

    while true; do
        case $1 in
            --)
                shift
                break
                ;;
            *)
                shift
```

```
                ;;
        esac
    done

    if [[ -n "$*" ]]; then
        printf  "$display_name: $*: %s\n$usage_line\n" \
                "$(gettext 'arguments are not allowed')"
        exit $error
    fi
}


function process_input {
    if groups "$USER" | grep --quiet --regexp='sudo'; then
        sudo true
    else
        printf  '%s\n' "$(gettext 'Already performed by the administrator.')"
        term_script
    fi
    printf "${bold}%s${normal} ($display_name)\n" "$(gettext 'Check')"
    check_dependencies
    printf "\n${bold}%s${normal} ($display_name)\n" "$(gettext 'Build')"
    pull_repos
    download_website
}


function check_dependencies {
    local escape='gettext'

    printf '%s\n' "$(gettext 'Check dependencies...')"
    # Install ghostscript for kz-build <man-pag>.pdf (ps2pdf).
    check_dpkgd_snapd
    sudo    apt-get             \
            install             \
            --yes               \
            curl                \
            fakeroot            \
            $escape             \
            ghostscript         \
            git                 \
            jq                  \
            lftp                \
            nmap                \
            pycodestyle         \
            python3-pycodestyle \
            python3-autopep8    \
            python3-pip         \
            python-is-python3
    sudo ln --force --relative --symbolic /usr/bin/pycodestyle /usr/bin/pep8
    sudo ln --force --relative --symbolic /usr/bin/pip3 /usr/bin/pip
    check_dpkgd_snapd
    # Debian package shellcheck is old, snap is newer.
    sudo snap install shellcheck
    check_dpkgd_snapd
    sudo snap install --classic code
}


function pull_repos {
    local bin_repo=/home/"$USER"/bin

    git config --global user.name 'Karel Zimmer'
    git config --global user.email 'karel.zimmer@gmail.com'
```

```
    git config --global pull.ff only
    git config --global credential.helper store

    if [[ -d "$bin_repo" ]]; then
        cd "$bin_repo"
        git pull
    else
        git clone https://github.com/karelzimmer/bin.git "$bin_repo"
    fi
    printf '%s\n' "$(gettext 'Pull repos (gitpull)...')"
    # Call gitpull.
    "$bin_repo"/gitpull
    printf '%s\n' "$(gettext 'Status repos (gitstat)...')"
    # Call gitstat.
    "$bin_repo"/gitstat
}


function download_website {
    local ftp_set='set ssl:verify-certificate no'
    local ftp_from=/httpdocs
    local ftp_to=$HOME/kz-uploads/dist
    local ftp_opts='--delete --verbose'
    local ftp_exclude='--exclude icaclient_20.04.0.21_amd64.deb'
    local ftp_cmd="mirror $ftp_exclude $ftp_opts $ftp_from $ftp_to; exit"
    local ftp_host=server106.hosting2go.nl
    local ftp_user=kzimmer
    local ftp_login=$HOME/.kz-$ftp_host

    printf '%s\n' "$(gettext 'Download website (lftp)...')"
    if ! [[ -f $ftp_login ]]; then
        read -rsp "$(gettext 'Password for') ftp://$ftp_host': " < /dev/tty
        printf '%s' "$REPLY" > "$ftp_login"
        printf '\n'
        chmod 'u=rw,g=,o=' "$ftp_login"
    fi
    if ! lftp   --user "$ftp_user"                  \
                --password "$(cat "$ftp_login")"    \
                -e "$ftp_set; $ftp_cmd"             \
                "$ftp_host"; then
        rm  "$ftp_login"
        printf '%s\n' "$(gettext 'Website download failed.')"
        return  1
    fi
}


function term_script {
    exit $ok
}


################################################################################
# Main
################################################################################

function main {
    init_script "$@"
    check_input "$@"
    process_input
    term_script
}

main "$@"
```