```
#!/bin/bash

# Too many commands will need superuser rights, so demand root!
if [ "$USER" != "root" ]; then
        echo "You must run this script as root."
        exit
fi

# Unmount the floppy just in case it's mounted
sync;sync
umount -r /dev/fd0 2> /dev/null
umount -l /dev/fd0 2> /dev/null

# Unmount the tempfs mount point just in case it was left mounted by script crash
umount -r tempfs 2> /dev/null
umount -l tempfs 2> /dev/null

# Do a low-level disk check.
badblocks -n -s /dev/fd0
if [ $? != 0 ]; then
        echo "Your floppy is bad!"
        exit
fi

echo "Formatting floppy..."
mformat a:

# Delete old loopback file system if it exists
if [ -e fs ]; then
        echo "Deleting old loopback file system"
        rm -rf fs
fi
if [ -e filesys ]; then
        echo "Deleting old compressed file system"
        rm -f filesys
fi

# Create mount point
```

```
echo "Creating 'tempfs' temporary mount point folder"
rm -r tempfs 2> /dev/null
mkdir tempfs

# Create empty loopback filesystem - 4MB - depending on how you count it
echo "Creating 4MB empty file for the loopback file system"
dd if=/dev/zero of=fs bs=1k count=4000 > /dev/null

# Create an ext2 file system in the loopback file with 2000 i-nodes (more than needed)
echo "Making new EXT2 file system in the 'fs' loopback file"
mke2fs -q -F -m 0 -N 2000 fs

# Mount the new empty loopback filesystem
echo "Mounting loopback file './fs' in the './tempfs/' directory"
LOOP_NUMBER=-1
false
until [ $? = 0 ]; do
        LOOP_NUMBER=$(($LOOP_NUMBER + 1))
        mount fs tempfs -o loop=/dev/loop$LOOP_NUMBER
done

# Create directories in the filesystem
echo "Creating directories in the filesystem"
for NEW_DIR in bin boot dev etc lib media mnt opt proc sbin tmp usr var root srv home sys; do
        mkdir tempfs/$NEW_DIR
done
mkdir tempfs/etc/init.d
mkdir tempfs/usr/bin
mkdir tempfs/usr/sbin
for NEW_DIR in cdrom floppy usb; do
        for MNT_DIR in mnt media; do
                mkdir tempfs/$MNT_DIR/$NEW_DIR
        done
done

# Create device files in the new filesystem
echo "Copying device files to the new filesystem"
cp -dpR /dev/fd[01]* tempfs/dev
```

```
cp -dpR /dev/tty[0-6] tempfs/dev
cp -dpR /dev/hd[a-d][0-20] tempfs/dev
cp -dpR /dev/sd[a-d][0-20] tempfs/dev
for NEW_DEVICE in console kmem mem null ram0 zero; do
        cp -dpR /dev/$NEW_DEVICE tempfs/dev
done
mkdir tempfs/dev/pts

# etc/fstab
echo "Creating /etc/fstab"
>  tempfs/etc/fstab echo "# Entry format - <device> <mount point> <fs type> <mount options> <dump> <fsck>"
>> tempfs/etc/fstab echo "/dev/root  /            ext2     rw,noauto                0  1"
>> tempfs/etc/fstab echo "devpts     /dev/pts     devpts   defaults,gid=5,mode=620  0  0"
>> tempfs/etc/fstab echo "proc       /proc        proc     defaults                 0  0"
>> tempfs/etc/fstab echo "/dev/ram0  /            ext2     defaults                 0  0"

# etc/group
echo "root:x:0:"    > tempfs/etc/group
echo "daemon:x:1:" >> tempfs/etc/group
echo "bin:x:2:"    >> tempfs/etc/group
echo "sys:x:3:"    >> tempfs/etc/group
echo "adm:x:4:"    >> tempfs/etc/group
echo "tty:x:5:"    >> tempfs/etc/group
echo "disk:x:6:"   >> tempfs/etc/group
echo "utmp:x:43:"  >> tempfs/etc/group

# etc/inittab
echo "Creating /etc/inittab"
>  tempfs/etc/inittab echo "# Entry format - <id>:<runlevels>:<action>:<process>"
>> tempfs/etc/inittab echo "# id        == tty to run on, or empty for /dev/console"
>> tempfs/etc/inittab echo "# runlevels == ignored"
>> tempfs/etc/inittab echo "# action    == one of sysinit, respawn, askfirst, wait, and once"
>> tempfs/etc/inittab echo "# process   == program to run"
>> tempfs/etc/inittab echo "::sysinit:/bin/mount -o remount,rw /"
>> tempfs/etc/inittab echo "::sysinit:/bin/mount -t proc proc /proc"
>> tempfs/etc/inittab echo "::sysinit:/bin/mount -a"
>> tempfs/etc/inittab echo "::sysinit:/etc/init.d/rcS" # busybox default
>> tempfs/etc/inittab echo "::askfirst:/bin/sh" # busybox default
```

```
>> tempfs/etc/inittab echo "::ctrlaltdel:/sbin/reboot" # busybox default
>> tempfs/etc/inittab echo "::shutdown:/sbin/swapoff -a" # busybox default
>> tempfs/etc/inittab echo "::shutdown:/bin/umount -a -r" # busybox default
>> tempfs/etc/inittab echo "::restart:/sbin/init" # busybox default

# etc/passwd
echo "Creating /etc/passwd"
echo "root::0:0:root:/:/bin/sh" > tempfs/etc/passwd

# rcStart
echo "Creating /etc/init.d/rcS"
echo "#!/bin/sh"          > tempfs/etc/init.d/rcS
echo 'echo "Welcome to Linux!"' >> tempfs/etc/init.d/rcS
chmod +x tempfs/etc/init.d/rcS

# Set up BusyBox
if [ -e /bin/busybox ]; then
        # Install busybox
        echo "Copying busybox to /bin"
        cp -a /bin/busybox tempfs/bin/
        # Create a link to busybox to take the place of the linuxrc script
        ln -s /bin/busybox tempfs/linuxrc
        # Here's all the BusyBox commands
        BUSYBOX_COMMANDS="adjtimex ar basename cat chgrp chmod chown chroot chvt clear cmp cp cpio cut date dc dd deallocvt df
dirname dmesg dos2unix dpkg dpkg-deb du dumpkmap dutmp echo expr false fbset fdflush find free freeramdisk fsck.minix getopt
grep gunzip gzip halt head hostid hostname id ifconfig init insmod kill killall klogd length ln loadacm loadfont loadkmap logger
logname ls lsmod makedevs md5sum mkdir mkfifo mkfs.minix mknod mkswap mktemp more mount mt mv nc nslookup ping pivot_root
poweroff printf ps pwd rdate readlink reboot renice reset rm rmdir rmmod route rpm2cpio sed setkeycodes sh sleep sort stty
swapoff swapon sync syslogd tail tar tee telnet test tftp touch tr true tty umount uname uniq unix2dos update uptime usleep
uudecode uuencode watchdog wc wget which whoami xargs yes zcat ["
        # Create links for busybox using the existing system as a clue for destination
        echo "Creating busybox links"
        for BUSYBOX_COMMAND in $BUSYBOX_COMMANDS; do
                FOUND_A_HOME="no"
                for DEST_DIR in "bin" "sbin" "usr/bin" "usr/sbin"; do
                        if [ -e /$DEST_DIR/$BUSYBOX_COMMAND ]; then
                                ln -s /bin/busybox tempfs/$DEST_DIR/$BUSYBOX_COMMAND
                                FOUND_A_HOME="yes"
```

```
                        fi
                done
                # If we can't find a home for a command, put it in /bin
                if [ "$FOUND_A_HOME" = "no" ]; then
                        ln -s /bin/busybox tempfs/bin/$BUSYBOX_COMMAND
                fi
        done
fi

# # Copy any needed library files
# echo "Copying files to /lib"
# cp -a /lib/libc.so*                          tempfs/lib 2> /dev/null
# cp -a /lib/uClibc*.so                        tempfs/lib 2> /dev/null
# cp -a /lib/ld.so-1/d-link/ld-linux-uclibc.so* tempfs/lib 2> /dev/null
# cp -a /lib/ld.so-1/libdl/libdl.so*           tempfs/lib 2> /dev/null
# cp -a /lib/crypt/libcrypt.so*                tempfs/lib 2> /dev/null

# PAM setup (may not be needed depending on the kernel used)
echo "Creating a default PAM file at /lib/security/pam_permit.so"
if [ -e /lib/security/pam_permit.so ]; then
        cp -a /lib/security/pam_permit.so /
        echo "OTHER   auth       optional     /lib/security/pam_permit.so"  > tempfs/etc/pam.conf
        echo "OTHER   account    optional     /lib/security/pam_permit.so" >> tempfs/etc/pam.conf
        echo "OTHER   password   optional     /lib/security/pam_permit.so" >> tempfs/etc/pam.conf
        echo "OTHER   session    optional     /lib/security/pam_permit.so" >> tempfs/etc/pam.conf
fi

# NSS setup (may not be needed depending on the kernel used)
echo "Creating a default NSS file at /etc/nsswitch.conf"
echo "passwd:     files" >  tempfs/etc/nsswitch.conf
echo "shadow:     files" >> tempfs/etc/nsswitch.conf
echo "group:      files" >> tempfs/etc/nsswitch.conf
echo "hosts:      files" >> tempfs/etc/nsswitch.conf
echo "services:   files" >> tempfs/etc/nsswitch.conf
echo "networks:   files" >> tempfs/etc/nsswitch.conf
echo "protocols:  files" >> tempfs/etc/nsswitch.conf
echo "rpc:        files" >> tempfs/etc/nsswitch.conf
echo "ethers:     files" >> tempfs/etc/nsswitch.conf
```

```
echo "netmasks:   files" >> tempfs/etc/nsswitch.conf
echo "bootparams: files" >> tempfs/etc/nsswitch.conf
echo "automount:  files" >> tempfs/etc/nsswitch.conf
echo "aliases:    files" >> tempfs/etc/nsswitch.conf
echo "netgroup:   files" >> tempfs/etc/nsswitch.conf
echo "publickey:  files" >> tempfs/etc/nsswitch.conf

# Miscellaneous
echo "Creating /var/log, /var/run, and /mnt/var/run/utmp"
mkdir -p tempfs/mnt/var/{log,run}
touch tempfs/mnt/var/run/utmp

# Unmount and compress the filesystem
echo "Compressing filesystem"
sync;sync
umount -r tempfs 2> /dev/null
umount -l tempfs 2> /dev/null
dd if=fs bs=1k | gzip -q -9 > filesys

# Mount the floppy so we can copy files to it
echo "Mounting the floppy as an MSDOS/FAT disk."
mount -t msdos /dev/fd0 tempfs

# Copy the linux kernel to the floppy
if [ -e linux ]; then
        echo "Copying the 'linux' kernel to the floppy"
        cp linux tempfs
else
        echo "Couldn't find a kernel named 'linux' in the current directory,"
        echo "so I'm using the kernel on your hard drive. It's probably too big..."
        cp /boot/vm* tempfs/linux
fi

# Copy the file system to the floppy
echo "Copying the compressed filesystem to the floppy"
cp filesys tempfs

# Put a SysLinux configuration file on the floppy
```

```
echo "Creating a /syslinux.cfg configuration file on the floppy"
echo "append initrd=filesys root=/dev/ram0" > tempfs/syslinux.cfg

# Unmount the floppy
echo "Unmounting the floppy"
sync;sync
umount -r tempfs 2> /dev/null
umount -l tempfs 2> /dev/null

# Run syslinux to make the floppy bootable
echo "Running syslinux to make the floppy bootable"
syslinux /dev/fd0

# Remove the temporary files now that we're done with them
echo "Cleaning up (deleting) 'fs', 'filesys', and 'tempfs'"
rm fs
rm filesys
rm -r tempfs

# Done
echo "Done. At least if everything WORKED it's done!"
```