Creating Bootable USB Disk Layout

Using GPT

freebsd_gpt_disk_layout.txt

```
creating disk layout (gpt) for freebsd
- assuming disk is da0 (change accordingly for other designation)

0- clean existing partitions/slices

# gpart destroy -F da0

(manual op)
find total sector count using diskinfo
# diskinfo -v da0
backup/secondary gpt table is at the last 34 sector
# echo '<total> - 34' | bc
<offset>
# dd if=/dev/zero of=/dev/da0 bs=512 seek=<offset>

1- create gpt and bios boot scheme

# gpart create -s gpt da0
da0 created
# gpart add -t freebsd-boot -l gpboot -b 40 -s 1004K da0
da0p1 added
# gpart bootcode -b /boot/pmbr -p /boot/gptboot -i 1 da0
partcode written to da0p1
bootcode written to da0

2- create efi partition

# gpart add -t efi -l gpefiboot -s 127M da0
da0p2 added
# newfs_msdos /dev/da0p2
...

3- copy efi binary

# mount -t msdosfs /dev/da0p2 /mnt
# mkdir -p /mnt/EFI/BOOT
# cp /boot/boot1.efi /mnt/EFI/BOOT/
# umount /mnt

4- create partition/slice for root and swap

# gpart add -t freebsd-ufs -l gprootfs -s 14G da0
da0p3 added
# gpart add -t freebsd-swap -l gpswap da0
```

```
da0p4 added

5- format/prepare fs for root

# newfs -U /dev/da0p3
```

Using MBR (just in case... :p)

freebsd_mbr_disk_layout.txt

```
# gpart create -s mbr da0
# gpart bootcode -b /boot/mbr da0
# gpart add -t freebsd da0
# gpart set -a active -i 1 da0
# gpart create -s bsd da0s1
# gpart bootcode -b /boot/boot da0s1
```