Projeto de Kiosk interativo para a Beaglebone Black com Yocto Parte - 2

Na segunda fase desse tutorial do **kiosk interativo**, vamos ver como montar a imagem do **kiosk-manage**r. Se você não viu a parte 1 ou quer rever o que foi apresentado , clique aqui.

O **kiosk-manager** é o responsável por configurar as páginas e propagandas que serão apresentadas nos **web-kiosks.** Neste exemplo estamos usando um **web-kiosk** e um **kiosk-manager**, sendo ambos usando uma **Beaglebone Black** cada um presente na mesma rede **LAN** onde os endereços **IP** foram atribuídos por um servidor **DHCP**.

Gerando a imagem do kiosk-manager

Depois de ter feito os passos de montagem do ambiente de construção e geração da imagem do **web-kiosk** da primeira parte da série, vamos gerar a imagem do **kiosk-manager** com os seguintes comandos:

```
cd ~/yocto
. . kiosk-src/oe-init-build-env kiosk-build
bitbake core-image-kiosk-manager
```

Gravando a imagem do kiosk-manager

Iremos gravar um outro **microSD** com a imagem do **kiosk-manager.**

Para a gravação da imagem no **microSD card**, o mesmo deve ser **particionado** e **formatado** de acordo o padrão aceito pela placa. Para isso foi criado um script, que pode ser obtido com os seguintes comandos:

Sccipts

```
cd ~/yocto/
git clone https://github.com/henriqueprossi/beaglebone-black.git bbb-sdcard-prepare
```

Insira o **microSD card** no computador ou notebook (host), e descubra qual o **device node** criado pelo **sistema operacional**. Digite um dos seguintes comandos:

Descoberta do device node

```
dmesg
sudo fdisk -l
```

Caso, por exemplo, o **device node** criado seja /dev/sdb, use os comandos:

Formatação do microSD card

```
cd bbb-sdcard-prepare
cd scripts
chmod +x format_sd_card.sh
sudo ./format_sd_card.sh /dev/sdb
```

Quando o processo de **particionamento** e **formatação** concluir, duas **partições** no **microSD card** são criadas:

boot (FAT32);
rootfs (ext4).

Execute a "**montagem**" dessas duas **partições** no sistema de arquivos do **sistema host**. Caso você use uma distribuição **Ubuntu**, basta retirar e inserir novamente o **microSD card** no **conector** do **PC**. Sendo que, por exemplo, os pontos de montagem criados sejam /media/boot e /media/rootfs, as imagens geradas anteriormente são copiadas da seguinte forma para o **kiosk-manager**:

Cópia das imagens no microSD card:

```
cd ~/yocto/kiosk-build/tmp/deploy/images/beaglebone
cp MLO /media/boot
cp u-boot.img /media/boot
sudo tar xzf core-image-kiosk-manager-beaglebone.tar.gz -C /media/rootfs
```

Testando a imagem do kiosk-manager e web-kiosk

Remova o **microSD** do computador e insira-o na **Beaglebone Black,** conecte-a a um **roteador** pelo **cabo ethernet**. Não se esqueça de conectar a outra **Beaglebone Black** que fará o papel de **web-kiosk** no mesmo **roteador**, além de conectar também o cabo **HDMI**.

Para o **kiosk-manager** configurar as páginas e propagandas que serão exibidas no **web-kiosk**, temos que saber qual é seu **IP**. Para isso vamos conectar um cabo serial no **web-kiosk** e rodar o comando abaixo:

Cópia das imagens no microSD card:

```
ifconfig


eth0      Link encap:Ethernet  HWaddr C8:A0:30:C4:61:4C
          inet addr:192.168.0.101  Bcast:0.0.0.0  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1284 errors:0 dropped:0 overruns:0 frame:0
          TX packets:436 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:217748 (212.6 KiB)  TX bytes:50863 (49.6 KiB)
          Interrupt:56

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
```

```
UP LOOPBACK RUNNING  MTU:65536  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Podemos ver que a interface de rede ethernet (eth0) está com o IP: **192.168.0.101**.

Agora iremos conectar o **cabo serial**, na **Beaglebone Black kiosk-manager** e configurar as páginas que queremos que sejam exibidas no **web-kiosk**.

O script responsável por essa configuração é o "manage_kiosk.sh" que está localizado em "/usr/bin/manage_kiosk.sh" e contém a seguinte implementação:

```
#!/bin/sh

client=192.168.1.101

changescript=/usr/bin/change_midori_url.sh


REMOTE_ADS="
100;http://alteredqualia.com/canvasmol/#Penicillin
120;http://peterned.home.xs4all.nl/3d/
120;http://andrew-hoyer.com/experiments/walking/
120;http://ie.microsoft.com/testdrive/performance/fishbowl/
120;http://fff.cmiscm.com
"

LOCAL_VIDEOS="
134;file:///var/local/ads/html5video/video-yp1.html
134;file:///var/local/ads/html5video/multi-video.html
170;file:///var/local/ads/html5video/video-yp2.html
140;file:///var/local/ads/html5video/video-seagaul.html
63;file:///var/local/ads/html5video/video-bunny.html
193;file:///var/local/ads/html5video/video-linux1.html
62;file:///var/local/ads/www.shinydemos.com/hipster-dog/index.html
"

LOCAL_INTERACTIVE_ADS="
62;file:///var/local/ads/www.shinydemos.com/beach/index.html
32;file:///var/local/ads/www.shinydemos.com/inbox-attack/index.html
32;file:///var/local/ads/www.shinydemos.com/rock-piano/index.html
"

ADS=${REMOTE_ADS}
ADS=${LOCAL_VIDEOS}
#ADS=${LOCAL_INTERACTIVE_ADS}
```

```
for ad in ${ADS}
do
  duration=`echo ${ad} | cut -d\; -f1 `
  url=`echo ${ad} | sed "s/^[0-9]*;//"`
  echo ${client} ${duration}Seconds ${url}
  ssh -x root@${client} ${changescript} ${url}
  sleep ${duration}
done
```

Vamos mudar o **IP** do **web-kiosk** na linha **3** e adicionar um comentário na linha **33** para que ele comande o **web-kiosk** a exibir páginas da internet.

De acordo com o que foi apresentado, o arquivo "manage_kiosk.sh" deverá ficar assim:

```
#!/bin/sh

client=192.168.0.101

changescript=/usr/bin/change_midori_url.sh


REMOTE_ADS="
100;http://alteredqualia.com/canvasmol/#Penicillin
120;http://peterned.home.xs4all.nl/3d/
120;http://andrew-hoyer.com/experiments/walking/
120;http://ie.microsoft.com/testdrive/performance/fishbowl/
120;http://fff.cmiscm.com
"

LOCAL_VIDEOS="
134;file:///var/local/ads/html5video/video-yp1.html
134;file:///var/local/ads/html5video/multi-video.html
170;file:///var/local/ads/html5video/video-yp2.html
140;file:///var/local/ads/html5video/video-seagaul.html
63;file:///var/local/ads/html5video/video-bunny.html
193;file:///var/local/ads/html5video/video-linux1.html
62;file:///var/local/ads/www.shinydemos.com/hipster-dog/index.html
"

LOCAL_INTERACTIVE_ADS="
62;file:///var/local/ads/www.shinydemos.com/beach/index.html
32;file:///var/local/ads/www.shinydemos.com/inbox-attack/index.html
32;file:///var/local/ads/www.shinydemos.com/rock-piano/index.html
"

ADS=${REMOTE_ADS}
#ADS=${LOCAL_VIDEOS}
#ADS=${LOCAL_INTERACTIVE_ADS}

for ad in ${ADS}
do
```

```
  duration=`echo ${ad} | cut -d\; -f1 `
  url=`echo ${ad} | sed "s/^[0-9]*;//"`
  echo ${client} ${duration}Seconds ${url}
  ssh -x root@${client} ${changescript} ${url}
  sleep ${duration}
done
```

Note que a frente de cada **URL** há um número. Esse número representa o tempo em segundos que a página ficará sendo mostrada. Se quiser, você pode alterar esses valores, bem como as *URLs*.

Para controlarmos o **web-kiosk** basta executarmos o comando:

```
manage_kiosk.sh
```

Na primeira vez que esse comando for executado o **ssh** pedirá a confirmação que se deseja conectar ao dispositivo e isso não ocorrerá mais nas próximas vezes.

Ate o próximo tutorial galera!