



Das enigma2-plugin Tutorial

von emanuel@i-have-a-dreambox.com

deutsche Ausgabe 1 * Stand: 14.02.2010

Vorab: Das Plugin Tutorial ist für **fortgeschrittene Benutzer** gedacht. Wenn Du mit den Begriffen in den Punkten unten nichts anfangen kannst, solltest du dich mal in unsere [>>> gp-wiki <<](#) umschauen.

Wenn du kein Grundtutorial in Python durchgemacht hast kannst du mal [>>> hier <<<](#) schauen.

Das brauchst du:

1. Unix Editor (z.B. GNU notepad++.exe (Mswin)
2. Ftp Verbindung zur Dreambox
3. ein Terminal (telnet/ssh) zur Dreambox
4. Grundkenntnis der Dreambox **BusyBox**
5. Grundtutorial in Python (am besten auf der Dreambox machen)
6. **lhad Tutorial plugin installieren, das Tutorial ist dann da:**
/usr/lib/enigma2/python/Plugins/lhadTutorial/
7. usr/lib/enigma2/python/Plugins/lhadTutorial/doc/doc_enigma2 bitte auch mal durchlesen! (Originale aus dem enigma2 CVS)
8. viel Spass, Freude und ein wenig Gedult

**Bitte keine Fragen zu den Grundvoraussetzungen
es geht hier um enigma2 plugins!**

Inhaltsverzeichnis:

- "Our Small Test" - ein reines Print Beispiel ohne Gui (OSD) S. 3-5
- lesson "01 Hallo World" – einfaches Fenster (Screen) S. 6-7
- lesson "02 Hallo World Message" – Screen mit Meldung S. 8-9
- lesson "03 Call My Msg" - Screen mit Meldung Ja/Nein S. 10-11
- lesson "04 My Menulist" – Screen mit Menuliste S. 12-14
- lesson "05 My Shell Prombt" – Screen mit Menuliste und Shell-kommandos, Ausgabe auf der Screen Konsole S. 15-17
- lesson "06 Message Input" – Screens mit Zeicheneingabe und Ausgabe S. 18-20
- lesson "07 View a picture" - Screen mit Bild S. 21-23
- lesson "08 Download a picture" - Download eines Bildes mit Ausgabe auf einem Screen S. 24-25
- lesson "09 dynamic Text" - Textlabel eines Screens ändern S. 26-27
- lesson "10 Set Auto Sleep" – Screen für Konfiguration des Startverhaltens der Dreambox S. 28-30
- lesson "11 Start other plugin" – Screen zum Starten des Bildbetrachters S. 31-32

So nachdem du ein **Python Grundtutorial** durch gearbeitet hast, kann es ans Eingemachte gehen...

Our Small Test

Wenn du das Tutorial plugin installiert hast, startet mal eine **Telnetsitzung** zur Dreambox und gibt das ein:

code:
root@dm8000:~# init 4; sleep 4; enigma2

Enigma2 startet neu, und du kannst im Telnet die enigma2 Meldungen sehen. Alle "print" Ausgaben oder Fehler (Crash), die dein plugin macht, werden dir hier angezeigt. (Auch schon beim Neustart von enigma2 bei einem plugin Quelltextfehler!!)

Die Telnetsitzung ist beim Entwickeln unser ständiger Begleiter!

Starte das minimal plugin Beispiel (es hat keine gui) aus dem enigma2 CVS doc, um das ganze mal zu testen.
(/usr/lib/enigma2/python/Plugins/lhadTutorial/OurSmallTest/..)

Öffnen den Plugin Browser auf der Dreambox und wähle das Plugin:



Beobachte die Ausgabe im Telnet:

```
code:
...
hdd IDLE!
[IDLE] 251.999181986 120 True
action -> WizardActions ok

<<<<<<<<<< Hello world! >>>>>>>>>

[EPGC] start cleanloop
[EPGC] stop cleanloop
...
```

Achtung wichtig!

Um den **Ausgabemodus** zu beenden drücke die Tastenkombination **[Strg c]** und enigma2 beendet sich.

```
code:
...
- (10) gRC
waiting for gRC thread shutdown
gRC thread has finished
- (9) gLCDDC
- (9) GFBDC
- (9) Font Render Class
- (8) graphics acceleration manager
- (1) Background File Eraser
reached rl -1
close frontend 0
root@dm8000:~#
```

Bitte warte immer bis unten die leere Telneteingabe erscheint denn enigma2 speichert noch!!

Um wieder enigma2 zu starten:

```
code:
root@dm8000:~# init 4; sleep 4; init 3
```

Der Quelltext zu Our Small Test:

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # "Our Small Test" - taken from enigma2/doc
3
4 from Plugins.Plugin import PluginDescriptor
5
6 def main(session, **kwargs):
7     print "\n<<<<<<<<< Hello world! >>>>>>>>\n"
8
9 def Plugins(**kwargs):
10     return PluginDescriptor(
11         name="Our Small Test",
12         description="plugin to test some capabilities",
13         where = PluginDescriptor.WHERE_PLUGINMENU,
14         icon="../ihad_tut.png",
15         fnc=main)
16
```

Achte auf den Modulimport:

aus **Plugins.Plugin** (Datei) /usr/lib/enigma2/python/Plugins/Plugin.py

wird **PluginDescriptor** geladen. Bitte schau immer in den Moduldateien, die du importierst nach, um zu wissen was du da lädst bzw. welche Parameter du brauchst. Am Anfang sieht es etwas unübersichtlich aus, aber das wird schon mit der Zeit.

In unserem Fall ist es das Modul, dass dafür sorgt das Plugin im Plugin Browser anzuzeigen. Wie Du in der Moduldatei siehst, gibt es viele, **PluginDescriptor** mehr dazu später in lesson 10.

Info:

Enigma2 Plugins können direkt auf der Box erstellt und geändert werden. Um Änderungen wirksam zu machen muss enigma2 neu gestartet werden (siehe oben)

Wenn du eigene Plugins erstellst sollen die im Verzeichnis:

/usr/lib/enigma2/python/Plugins/Extensions/<dein Pluginordner> z.B. "MeinPlugin" erstellen.

Das Tutorial ist nur der Übersicht wegen wo anders!

Drin muss sein:

eine Datei **__init__.py** (kann leer sein) und eine Datei **plugin.py** (mit dem Quelltext).

01 Hallo World

Ist ein einfaches Fenster (Screen) mit einem Textlabel und Beenden Kommando.

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 1
3 # by emanuel
4 #####
5
6 from Screens.Screen import Screen
7 from Components.Label import Label
8 from Components.ActionMap import ActionMap
9 from Plugins.Plugin import PluginDescriptor
10
11 #####
12
13 class HalloWorldScreen(Screen):
14     skin = """
15         <screen position="130,150" size="460,150" title="Ihad.tv e2-tutorial
16 lesson 1" >
17             <widget name="myLabel" position="10,60" size="200,40"
18 font="Regular;20"/>
19             </screen>"""
20
21     def __init__(self, session, args = None):
22         self.session = session
23
24         Screen.__init__(self, session)
25         self["myLabel"] = Label("Hello World ;-)")
26         self["myActionMap"] = ActionMap(["SetupActions"],
27 {
28     "cancel": self.close # add the RC Command "cancel" to close
29 your Screen
30 }, -1)
31
32 #####
33
34 def main(session, **kwargs):
35     print "\n[Hallo World] start\n"
36
37     session.open(HalloWorldScreen)
38
39 #####
40
41 def Plugins(**kwargs):
42     return PluginDescriptor(
43         name="01 Hallo World",
44         description="lesson 1 - Ihad.tv e2-tutorial",
45         where = PluginDescriptor.WHERE_PLUGINMENU,
46         icon="../ihad_tut.png",
47         fnc=main)
```

In lesson 01 siehst Du wie man eine Screen Klasse baut. Was ist alles nötig?

- 1) imports Zeile 6-9 (beachte wieder die Dateien dahinter!!)
(/usr/lib/enigma2/python/...)
- 2) Screen Klasse **HalloWorldScreen** Zeile 13-27
- 3) main Funktion (startet die HalloWorldScreen) Zeile 32-35
- 4) PluginDescriptor Zeile 39-45

Erklärung HalloWorldScreen Klasse:

in Zeile 14-17 wird das skin im xml format festgelegt. 1 screen, 1 widget (Textlabel)
in Zeile 19-27 wird die initialisierung der Klasse definiert.

zu Zeile 19:

__init__ wird beim Pluginstart aufgerufen (Parameter beachten). Eine Screen braucht als 1. Parameter immer **self** und als 2. Parameter immer eine session! Im unserem Bsp. wird der Parameter session von der main Funktion geliefert.

zu Zeile 20:

Der durch **__init__** (siehe oben) gelieferte Parameter **session** wird auf die **klasseninternen globale Variable self.session** abgespeichert für die weitere Verwendung.

zu Zeile 22:

initialisierungs Funktion der Screen Klasse mit den Parametern **self, self.session** aufrufen.

zu Zeile 23:

das in Zeile 16 mit Attributen festgelegte **widget "myLabel"** wird hier als **Label** mit einem festen Text beschrieben.

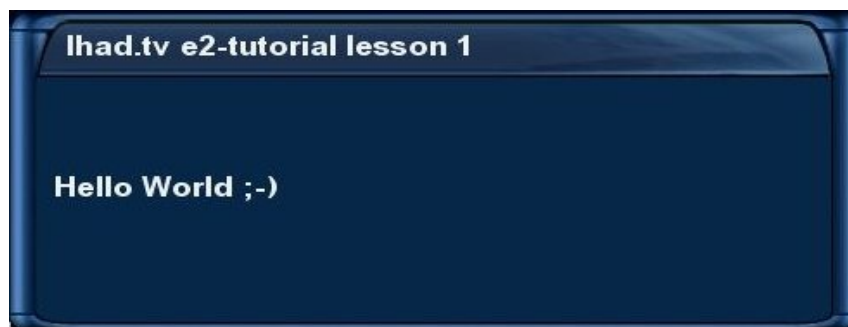
zu Zeile 24-27:

hier wird unsere ActionMap festgelegt. In unserem Fall die minimal Belegung "Screen schliessen". In Zeile 26 wird die Screen Funktion **self.close** dem Fenbediehnungskommando "**cancel**" aus SetupActions (FB "exit") zugewiesen.

beachte:

Die **ActionMap** "SetupActions" ist in:
/usr/share/enigma2/keymap.xml

Ergebnis:



02 Hallo World Message

Eine Screen mit Meldung.

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 2
3 # by emanuel
4 from Screens.Screen import Screen
5 from Components.Label import Label
6 from Components.ActionMap import ActionMap
7 from Screens.MessageBox import MessageBox
8 from Plugins.Plugin import PluginDescriptor
9
10 #####
11
12 class HalloWorldMsg(Screen):
13     skin = """
14         <screen position="130,150" size="460,150" title="Ihad.tv e2-tutorial
15         lesson 2" >
16             <widget name="myLabel" position="10,60" size="200,40"
17             font="Regular;20"/>
18             </screen>"""
19
20     def __init__(self, session, args = 0):
21         self.session = session
22         Screen.__init__(self, session)
23
24         self["myLabel"] = Label(_("please press OK"))
25         self["myActionMap"] = ActionMap(["SetupActions"],
26         {
27             "ok": self.myMsg,
28             "cancel": self.cancel
29         }, -1)
30
31     def myMsg(self):
32         print "\n[HalloWorldMsg] OK pressed \n"
33         self.session.open(MessageBox, _("Hello World!"), MessageBox.TYPE_INFO)
34
35     def cancel(self):
36         print "\n[HalloWorldMsg] cancel\n"
37         self.close(False, self.session)
38
39 #####
40
41 def main(session, **kwargs):
42     print "\n[HalloWorldMsg] start\n"
43     session.open(HalloWorldMsg)
44
45 #####
46
47 def Plugins(**kwargs):
48     return PluginDescriptor(
49         name="02 Hallo World Message",
50         description="lesson 2 - Ihad.tv e2-tutorial",
51         where = PluginDescriptor.WHERE_PLUGINMENU,
52         icon="../ihad_tut.png",
53         fnc=main)
```

Erklärung HalloWorldMsg Klasse:

Als Ergänzungen gegenüber lesson 01 solltest du dir hier die Zeile 7, 25-26 , 29-35 ansehen.

zu Zeile 7:

import der Screen Klasse **MessageBox**
(/usr/lib/enigma2/python/Screens/MessageBox.py)

zu Zeile 25-26:

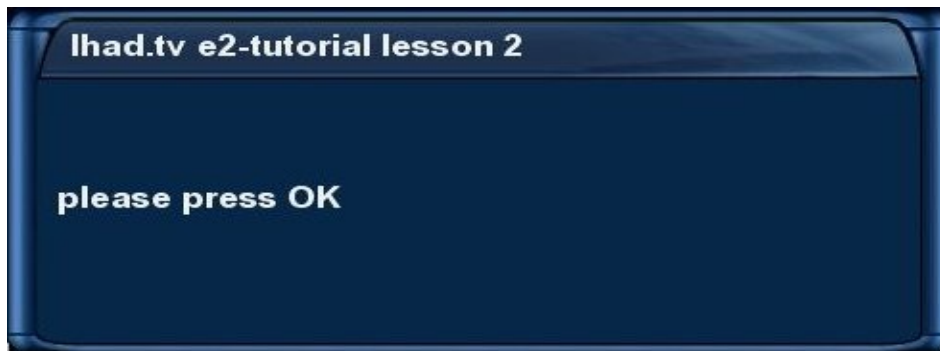
hier wird unsere ActionMap die in Zeile 29-35 definierten Klassenfunktionen zugewiesen.

zu Zeile 29-35:

in "def myMsg" wird die Screen MessageBox aufgerufen.

in "def cancel" wird die normale **self.close** mit Parametern aufgerufen
(nur zu Bsp. Zweck)

Ergebnis:



03 Call My Msg

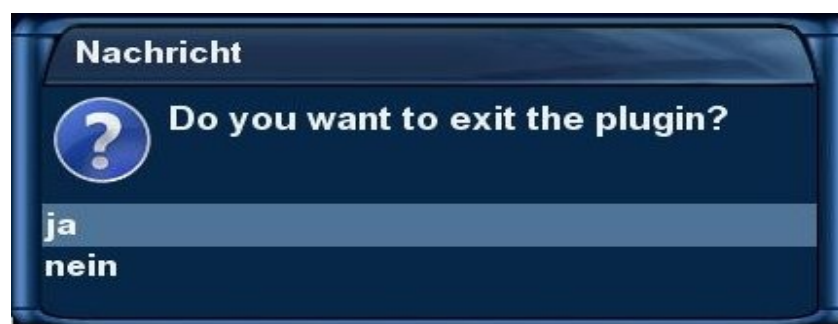
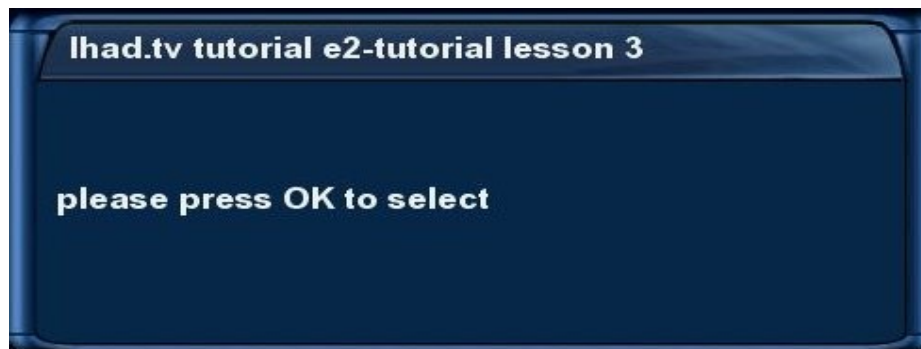
Eine Screen mit Meldung Ja/Nein (Funktion Screen schliesen).

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 3
3 # by emanuel
4 from Screens.Screen import Screen
5 from Components.Label import Label
6 from Components.ActionMap import ActionMap
7 from Screens.MessageBox import MessageBox
8 from Plugins.Plugin import PluginDescriptor
9
10 #####
11
12 class CallMyMsg(Screen):
13     skin = """
14         <screen position="130,150" size="460,150" title="Ihad.tv tutorial e2-
15 tutorial lesson 3" >
16             <widget name="myLabel" position="10,60" size="400,120"
17 font="Regular;20"/>
18             </screen>"""
19
20     def __init__(self, session, args = 0):
21         self.session = session
22         Screen.__init__(self, session)
23
24         self["myLabel"] = Label(_("please press OK to select"))
25         self["myActionMap"] = ActionMap(["SetupActions"],
26 {
27     "ok": self.myMsg,
28     "cancel": self.cancel
29 }, -1)
30
31     def callMyMsg(self, result):
32         print "\n[CallMyMsg] checking result\n"
33         if result:
34             print "\n[CallMyMsg] cancel\n"
35             self.close(None)
36         else:
37             self.session.open(MessageBox, _("Ah, you like the Ihad
38 plugin!\n;-)"), MessageBox.TYPE_INFO)
39
40     def myMsg(self):
41         print "\n[CallMyMsg] OK pressed \n"
42         self.session.openWithCallback(self.callMyMsg, MessageBox, _("Do you
43 want to exit the plugin?"), MessageBox.TYPE_YESNO)
44
45     def cancel(self):
46         print "\n[CallMyMsg] cancel\n"
47         self.close(None)
```

Fortsetzung src zu: CallMyMsg

```
45 #####
46
47 def main(session, **kwargs):
48     print "\n[CallMyMsg] start\n"
49     session.open(CallMyMsg)
50
51 #####
52
53 def Plugins(**kwargs):
54     return PluginDescriptor(
55         name="03 Call My Msg",
56         description="lesson 3 - Ihad.tv e2-tutorial",
57         where = PluginDescriptor.WHERE_PLUGINMENU,
58         icon="../ihad_tut.png",
59         fnc=main)
60
```

Ergebnis:



04 My Menulist

Eine Screen mit Menu.

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 4
3 # by emanuel
4 from Screens.Screen import Screen
5 from Components.MenuList import MenuList
6 from Components.ActionMap import ActionMap
7 from Screens.MessageBox import MessageBox
8 from Plugins.Plugin import PluginDescriptor
9
10 #####
11
12 class MyMenu(Screen):
13     skin = """
14         <screen position="100,150" size="460,400" title="Ihad.tv tutorial e2-
15 tutorial lesson 4" >
16             <widget name="myMenu" position="10,10" size="420,380"
17 scrollbarMode="showOnDemand" />
18         </screen>"""
19
20     def __init__(self, session, args = 0):
21         self.session = session
22
23         list = []
24         list.append(("Entry 1"), "one")
25         list.append(("Entry 2"), "two")
26         list.append(("Entry 3"), "tree")
27         list.append(("Exit"), "exit")
28
29         Screen.__init__(self, session)
30         self["myMenu"] = MenuList(list)
31         self["myActionMap"] = ActionMap(["SetupActions"],
32 {
33     "ok": self.go,
34     "cancel": self.cancel
35 }, -1)
36
37     def go(self):
38         returnValue = self["myMenu"].l.getCurrentSelection()[1]
39         print "\n[MyMenu] returnValue: " + returnValue + "\n"
40
41         if returnValue is not None:
42             if returnValue is "one":
43                 self.myMsg("1")
44
45             elif returnValue is "two":
46                 self.myMsg("2")
47
48             elif returnValue is "tree":
49                 self.myMsg("3")
50
51             else:
52                 print "\n[MyMenu] cancel\n"
53                 self.close(None)
```

Fortsetzung src zu: MyMenu

```
53     def myMsg(self, entry):
54         self.session.open(MessageBox, _("You selected entry no. %s!")
55 % (entry), MessageBox.TYPE_INFO)
56
57     def cancel(self):
58         print "\n[MyMenu] cancel\n"
59         self.close(None)
60
61 #####
62
63 def main(session, **kwargs):
64     print "\n[MyMenu] start\n"
65     session.open(MyMenu)
66
67 #####
68
69 def Plugins(**kwargs):
70     return PluginDescriptor(
71         name="04 My Menulist",
72         description="lesson 4 - Ihad.tv e2-tutorial",
73         where = PluginDescriptor.WHERE_PLUGINMENU,
74         icon="../ihad_tut.png",
75         fnc=main)
76
```

Erklärung MyMenu Klasse:

zu Zeile 5:

import der Menuliste. Datei zu Components.MenuList bitte ansehen!
(/usr/lib/enigma2/python/Components/MenuList.py)

zu Zeile 15:

im Screen Skin wird hier nur ein widget für die Menuliste festgelegt.

zu Zeile 21-25:

die python Liste für die Menulist wird gebaut.

zu Zeile 28:

dem **widget "myMenu"** wird hier die **MenuList**, mit der in Zeile 21-25 erzeugten Liste als Parameter, zugewiesen.

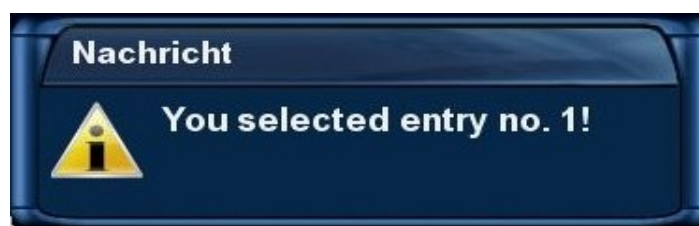
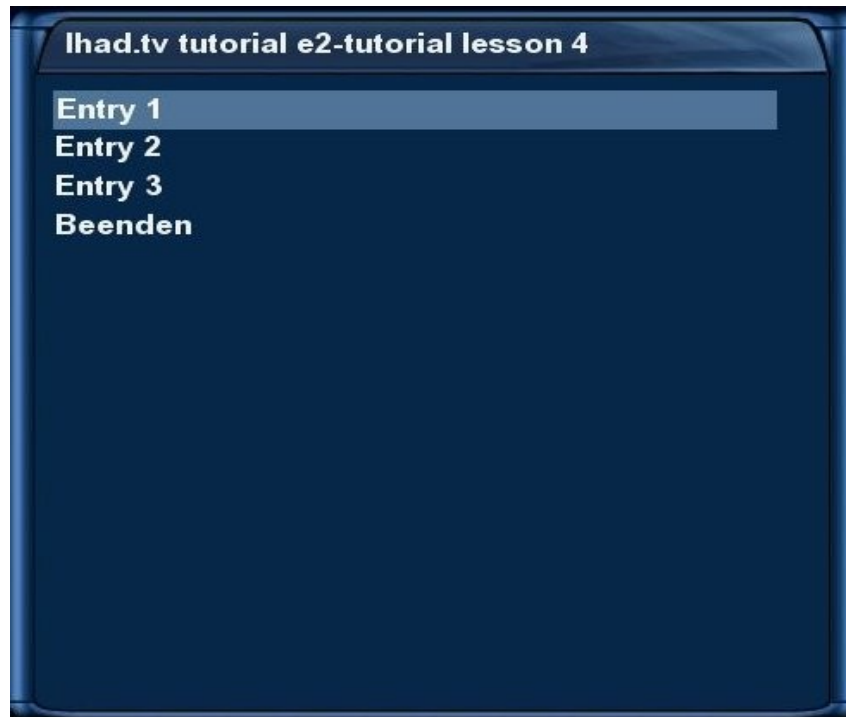
zu Zeile 31:

FB Kommando "ok" wird der Funktion "self.go" zugewiesen

zu Zeile 35-51:

self["myMenu"].l.getCurrentSelection()[1] liefert den zweiten Eintrag eines mit FB OK ausgewählten Listeneintrags. Bsp: **(_("Entry 1"), "one") => "one"**

Ergebnis:



05 My Shell Prombt

Eine Screen mit Menu Liste zum Starten von Shellkommandos

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 5 - copyrights 2010 by emanueel@ihad.tv
3 # by emanuel
4 from Screens.Screen import Screen
5 from Screens.Console import Console
6 from Components.MenuList import MenuList
7 from Components.ActionMap import ActionMap
8 from Plugins.Plugin import PluginDescriptor
9
10 #####
11
12 class MyShPrombt(Screen):
13     skin = """
14         <screen position="100,150" size="460,400" title="Ihad.tv tutorial e2-
15 tutorial lesson 5" >
16             <widget name="myMenu" position="10,10" size="420,380"
17 scrollbarMode="showOnDemand" />
18         </screen>"""
19
20     def __init__(self, session, args = 0):
21         self.session = session
22
23         list = []
24         list.append(("netstat", "com_one"))
25         list.append(("ls -ls /", "com_two"))
26         list.append(("mount", "com_tree"))
27         list.append((_("Exit"), "exit"))
28
29         Screen.__init__(self, session)
30         self["myMenu"] = MenuList(list)
31         self["myActionMap"] = ActionMap(["SetupActions"],
32 {
33     "ok": self.go,
34     "cancel": self.cancel
35 }, -1)
36
37     def go(self):
38         returnValue = self["myMenu"].l.getCurrentSelection()[1]
39         print "\n[MyShPrombt] returnValue: " + returnValue + "\n"
40
41         if returnValue is not None:
42             if returnValue is "com_one":
43                 self.prombt("/bin/netstat")
44
45             elif returnValue is "com_two":
46                 self.prombt("/bin/ls -ls /")
47
48             elif returnValue is "com_tree":
49                 self.prombt("/bin/mount")
50
51             else:
52                 print "\n[MyShPrombt] cancel\n"
53                 self.close(None)
```

Fortsetzung src zu: MyShPrombt

```
53     def go(self):
54         returnValue = self["myMenu"].l.getCurrentSelection()[1]
55         print "\n[MyShPrombt] returnValue: " + returnValue + "\n"
56
57         if returnValue is not None:
58             if returnValue is "com_one":
59                 self.prombt("/bin/netstat")
60
61             elif returnValue is "com_two":
62                 self.prombt("/bin/ls -ls /")
63
64             elif returnValue is "com_tree":
65                 self.prombt("/bin/mount")
66
67             else:
68                 print "\n[MyShPrombt] cancel\n"
69                 self.close(None)
70
71         def prombt(self, com):
72             self.session.open(Console, _("start shell com: %s") % (com), ["%s" %
73 com])
74
75         def cancel(self):
76             print "\n[MyShPrombt] cancel\n"
77             self.close(None)
78
79 #####
80
81 def main(session, **kwargs):
82     print "\n[MyShPrombt] start\n"
83     session.open(MyShPrombt)
84
85 #####
86
87 def Plugins(**kwargs):
88     return PluginDescriptor(
89         name="05 My Shell Prombt",
90         description="lesson 5 - Ihad.tv e2-tutorial",
91         where = PluginDescriptor.WHERE_PLUGINMENU,
92         icon="../ihad_tut.png",
93         fnc=main)
94
```

Erklärung MyShPrombt Klasse:

zu Zeile 5:
import der Console

zu Zeile 71-72:
“def prombt” führt die Konsolen Screen aus. Als Parameter bekommt sie das Shellkommando. Aufruf in Zeile: 41, 44, 47

Ergebnis:

```
lhad.tv tutorial e2-tutorial lesson 5
netstat
ls -ls /
mount
Beenden
```

```
start shell com: /bin/netstat
0:4038 ESTABLISHED
tcp      0      46 192.168.1.253:57754      p4FD9D149.dip.t-
di:6869 ESTABLISHED
tcp      0 16458 192.168.1.253:https     dslb-084-056-036-
0:4037 ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags      Type       State      I-Node Path
unix  2      [ ]       DGRAM          160858
@/org/kernel/udev/monitor
unix  2      [ ]       DGRAM          957
@/org/kernel/udev/udev
unix  3      [ ]       STREAM        CONNECTED  161022
/tmp/camd.socket
unix  3      [ ]       STREAM        CONNECTED  161021
unix  2      [ ]       STREAM        CONNECTED  160889
/tmp/gdaemon.socket
unix  3      [ ]       STREAM        CONNECTED  3785
unix  3      [ ]       STREAM        CONNECTED  3784
Ausführung beendet!
```

06 Message Input

Ein- und Ausgabe auf Screens

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 6
3 # by emanuel
4 from Screens.Screen import Screen
5 from Components.Label import Label
6 from Components.ActionMap import ActionMap
7 from Components.Input import Input
8 from Screens.InputBox import InputBox
9 from Screens.MessageBox import MessageBox
10 from Plugins.Plugin import PluginDescriptor
11
12 #####
13
14 class MsgInput(Screen):
15     skin = """
16     <screen position="130,150" size="460,150" title="Ihad.tv e2-tutorial
17     lesson 6" >
18         <widget name="myLabel" position="10,60" size="200,40"
19         font="Regular;20"/>
20     </screen>"""
21
22     def __init__(self, session, args = 0):
23         self.session = session
24         Screen.__init__(self, session)
25
26         self["myLabel"] = Label(_("please press OK"))
27         self["myActionMap"] = ActionMap(["SetupActions"],
28         {
29             "ok": self.myInput,
30             "cancel": self.cancel
31         }, -1)
32
33     def myInput(self):
34         self.session.openWithCallback(self.askForWord, InputBox,
35         title=_("Please enter a name for prombt!"), text=" " * 55, maxSize=55,
36         type=Input.TEXT)
37
38     def askForWord(self, word):
39         if word is None:
40             pass
41         else:
42             self.session.open(MessageBox,_(word), MessageBox.TYPE_INFO)
43
44     def cancel(self):
45         print "\n[MsgInput] cancel\n"
46         self.close(None)
```

Fortsetzung src zu: MsgInput

```
44 #####
45
46 def main(session, **kwargs):
47     print "\n[MsgInput] start\n"
48     session.open(MsgInput)
49
50 #####
51
52 def Plugins(**kwargs):
53     return PluginDescriptor(
54         name="06 Message Input",
55         description="lesson 6 - Ihad.tv e2-tutorial",
56         where = PluginDescriptor.WHERE_PLUGINMENU,
57         icon="../ihad_tut.png",
58         fnc=main)
59
```

Erklärung MsgInput Klasse:

die Klasse MsgInput ähnelt sehr stark der in lesson 3. Hier werden nur nicht 0 oder 1 als Ergebnis geliefert, sondern Eingaben der FB/Keyboard.

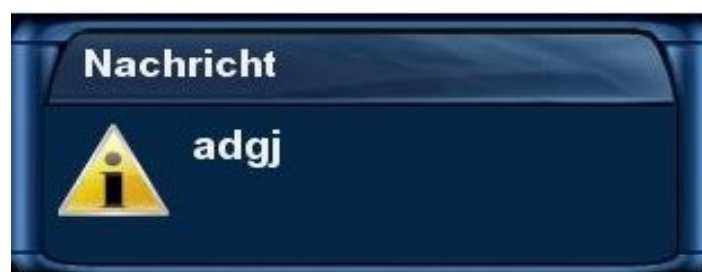
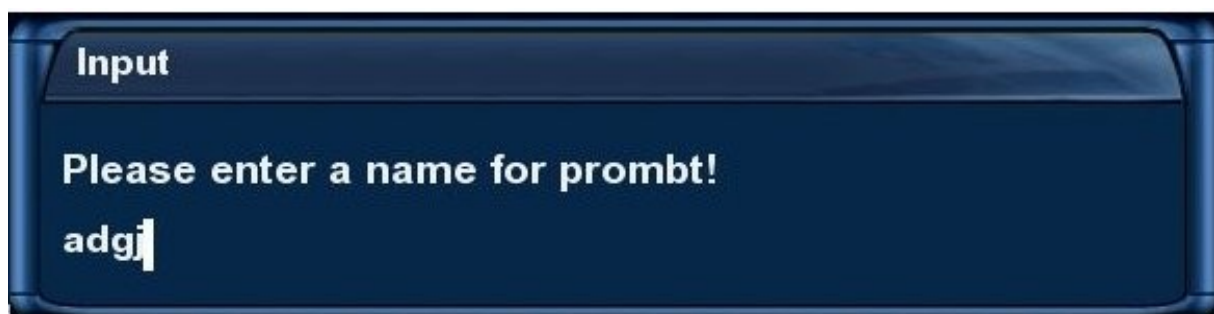
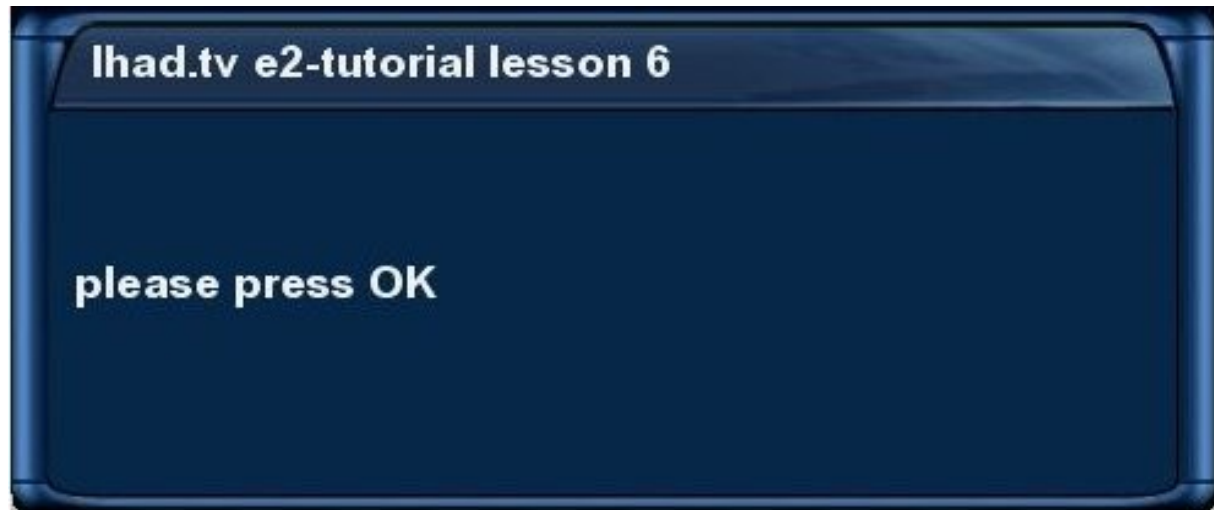
zu Zeile 7,8:

import von Input, InputBox

zu Zeile 31-38:

“def myInput” führt die InputBox Screen aus. openWithCallback liefert wieder das Ergebnis für “def askForWord” in Zeile aus der InputBox.

Ergebnis:



07 View a picture

Screen mit Bild

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 7
3 # by emanuel
4 from Screens.Screen import Screen
5 from Components.Label import Label
6 from Components.Pixmap import Pixmap
7 from Components.AVSwitch import AVSwitch
8 from Components.ActionMap import ActionMap
9 from Plugins.Plugin import PluginDescriptor
10 from enigma import ePicLoad
11
12 #####
13
14 class PictureScreen(Screen):
15
16     skin="""
17         <screen name="PictureScreen" position="0,0" size="720,576"
18         title="Picture Screen" backgroundColor="#002C2C39">
19             <widget name="myPic" position="0,0" size="720,576"
20             zPosition="1" alphatest="on" />
21         </screen>"""
22
23     def __init__(self, session, picPath = None):
24         Screen.__init__(self, session)
25         print "[PictureScreen] __init__\n"
26         self.picPath = picPath
27         self.Scale = AVSwitch().getFramebufferScale()
28         self.PicLoad = ePicLoad()
29         self["myPic"] = Pixmap()
30         self["myActionMap"] = ActionMap(["SetupActions"],
31         {
32             "ok": self.cancel,
33             "cancel": self.cancel
34         }, -1)
35
36         self.PicLoad.PictureData.get().append(self.DecodePicture)
37         self.onLayoutFinish.append(self.ShowPicture)
38
39     def ShowPicture(self):
40         if self.picPath is not None:
41             self.PicLoad.setPara([
42                 self["myPic"].instance.size().width(),
43                 self["myPic"].instance.size().height(),
44                 self.Scale[0],
45                 self.Scale[1],
46                 0,
47                 1,
48                 "#002C2C39"])
49             self.PicLoad.startDecode(self.picPath)
```

Fortsetzung src zu: PictureScreen

```
51     def DecodePicture(self, PicInfo = ""):
52         if self.picPath is not None:
53             ptr = self.PicLoad.getData()
54             self["myPic"].instance.setPixmap(ptr)
55
56
57     def cancel(self):
58         print "[PictureScreen] - cancel\n"
59         self.close(None)
60
61     #####
62
63     def main(session, **kwargs):
64         session.open(PictureScreen, picPath =
65         "/usr/share/enigma2/skin_default/icons/dish.png")
66
67     #####
68     def Plugins(**kwargs):
69         return PluginDescriptor(
70             name="07 View a picture",
71             description="lesson 7 - Ihad.tv e2-tutorial",
72             where = PluginDescriptor.WHERE_PLUGINMENU,
73             icon="../ihad_tut.png",
74             fnc=main)
75
```

Erklärung PictureScreen Klasse:

die Klasse PictureScreen ist ein Beispiel wie man ein Bild darstellt in einem Screen.

zu Zeile 6,7,9:

import von QPixmap, AVSwitch, ePicLoad beachten!

zu Zeile 18:

im Screen Skin wird ein **widget** "myPic" für das Bild festgelegt.

zu Zeile 21:

die Screen Klasse PictureScreen bekommt in `__init__` einen zusätzlichen Parameter "picPath" Aufruf in Funktion "main" Zeile 64.

zu Zeile 26:

Speicher fürs Bild

zu Zeile 27:

dem **widget** `self["myPic"]` wird eine QPixmap() ohne Parameter zugewiesen!

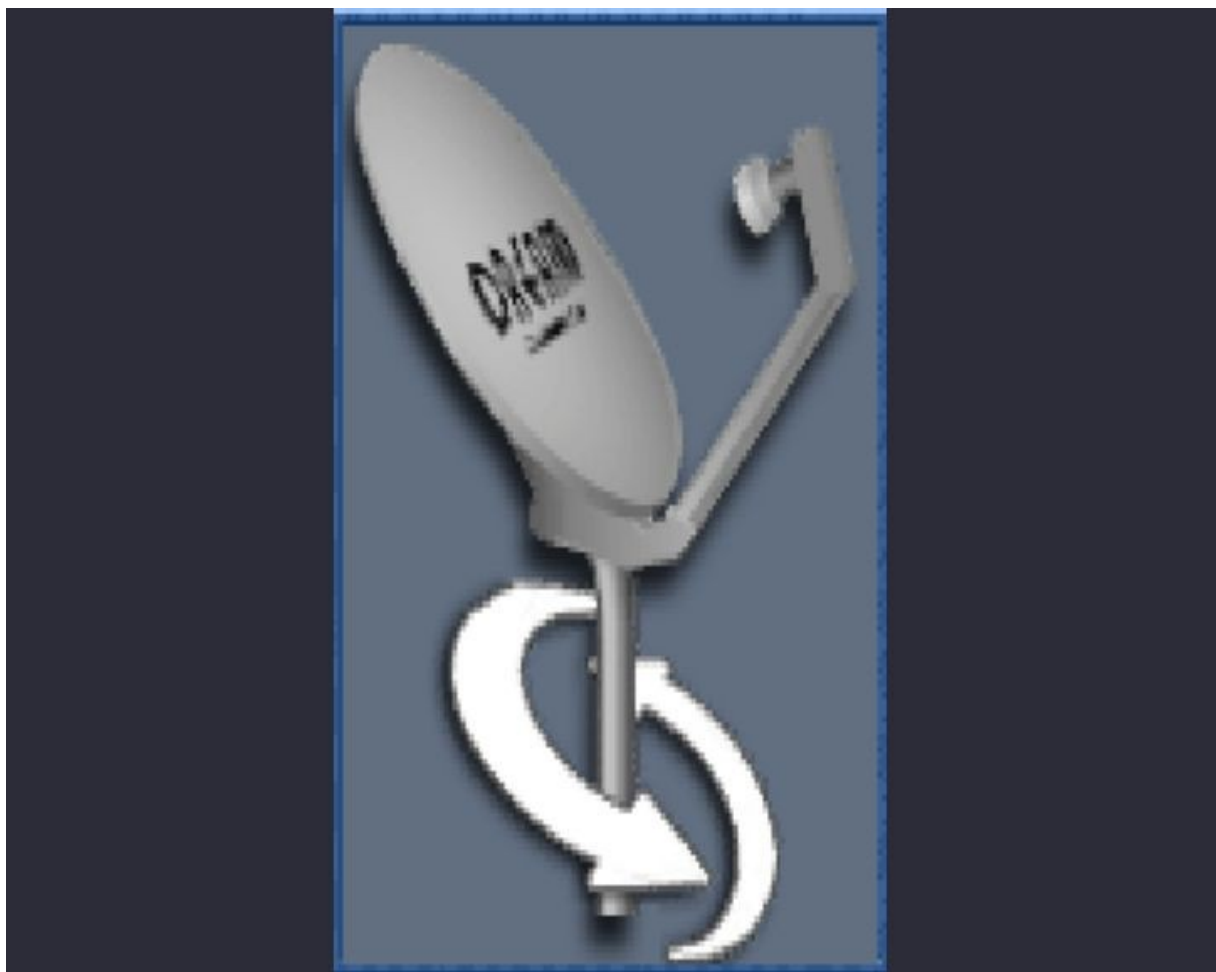
zu Zeile 34:

Laden und decodieren des Bildes; siehe Zeile 51-54

zu Zeile 35:

die Screen Funktion "onLayoutFinish.append([self.ShowPicture](#))" ist um das laden des Bildes bei Fertigstellung des Screenlayout anzuzeigen. Siehe Zeile 37-48, da werden die Parameter für das Bild gesetzt.

Ergebnis:



08 Download a picture

Download eines Bildes mit Ausgabe auf einem Screen aus lesson 07

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 8
3 # by emanuel
4 #####
5
6 from twisted.web.client import downloadPage
7 from Screens.MessageBox import MessageBox
8 from Plugins.IhadTutorial.lesson_07.plugin import PictureScreen
9 from Plugins.Plugin import PluginDescriptor
10
11 #####
12
13 class getPicfromUrl(object):
14     def __init__(self, session, url=None, path=None):
15         self.path = path
16         self.session = session
17         self.download(url, path)
18
19     def download(self, url, path):
20         downloadPage(url,
21 path).addCallback(self.downloadDone).addErrback(self.downloadError)
22
23     def downloadError(self, raw):
24         print "[e2Fetcher.fetchPage]: download Error", raw
25         self.session.open(MessageBox, text = _("Error downloading: ") +
26 self.path, type = MessageBox.TYPE_ERROR)
27
28     def downloadDone(self, raw):
29         print "[e2Fetcher.fetchPage]: download done", raw
30         self.session.open(PictureScreen, picPath = self.path)
31
32 #####
33 def main(session, **kwargs):
34     getPicfromUrl(session, "http://www.i-have-a-dreambox.com/images/ihad.jpg",
35 "/tmp/myPic.tmp")
36
37 #####
38 def Plugins(**kwargs):
39     return PluginDescriptor(
40         name="08 Download a picture",
41         description="lesson 8 - Ihad.tv e2-tutorial",
42         where = PluginDescriptor.WHERE_PLUGINMENU,
43         icon="../ihad_tut.png",
44         fnc=main)
```

Erklärung getPicfromUrl Klasse:

die Klasse getPicfromUrl importiert das Screen von lesson 07 um ein heruntergeladenes Bild darzustellen.

zu Zeile 6,8:

import von downloadPage,
PictureScreen aus lesson 07 (eigen)

zu Zeile 13-28:

die Klasse **getPicfromUrl(object)**: ist kein Screen; trotzdem braucht es um die **PictureScreen** bzw. bei einer Fehlermeldung die **MessageBox** den Parameter **"session"**! siehe Zeile 14 `__init__`

Ergebnis:



09 dynamic Text

Textlabel eines Screens ändern

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 9
3 # by emanuel
4 #####
5
6 from Screens.Screen import Screen
7 from Components.Label import Label
8 from Components.ActionMap import ActionMap
9 from Plugins.Plugin import PluginDescriptor
10
11 #####
12
13 class MyDynaTextScreen(Screen):
14     skin = """
15         <screen position="130,150" size="460,150" title="Ihad.tv e2-tutorial
16         lesson 9" >
17             <widget name="myText" position="10,50" size="400,40"
18             valign="center" halign="center" zPosition="2" foregroundColor="white"
19             font="Regular;22"/>
20             <widget name="myRedBtn" position="10,110" size="100,40"
21             backgroundColor="red" valign="center" halign="center" zPosition="2"
22             foregroundColor="white" font="Regular;20"/>
23             <widget name="myGreenBtn" position="120,110" size="100,40"
24             backgroundColor="green" valign="center" halign="center" zPosition="2"
25             foregroundColor="white" font="Regular;20"/>
26         </screen>"""
27
28     def __init__(self, session, args = 0):
29         self.session = session
30         Screen.__init__(self, session)
31
32         self.text="Press green or ok button to edit text!"
33         self["myText"] = Label()
34         self["myRedBtn"] = Label(_("Cancel"))
35         self["myGreenBtn"] = Label(_("OK"))
36         self["myActionsMap"] = ActionMap(["SetupActions", "ColorActions"],
37         {
38             "ok": self.editMyText,
39             "green": self.editMyText,
40             "red": self.close,
41             "cancel": self.close,
42         }, -1)
43         self.onShown.append(self.setMyText)
44
45     def setMyText(self):
46         self["myText"].setText(self.text)
47
48     def editMyText(self):
49         self.text="I love Ihad.tv!\n:-)"
50         self.setMyText()
```

Fortsetzung src zu: MyDynaTextScreen

```
45 #####
46
47 def main(session, **kwargs):
48     session.open(MyDynaTextScreen)
49
50 #####
51
52 def Plugins(**kwargs):
53     return PluginDescriptor(
54         name="09 dynamic Text",
55         description="lesson 9 - Ihad.tv e2-tutorial",
56         where = PluginDescriptor.WHERE_PLUGINMENU,
57         icon="../ihad_tut.png",
58         fnc=main)
59
```

Erklärung MyDynaTextScreen Klasse:

Das sollte für dich jetzt kein Problem mehr sein.

zu Zeile 26:

`self["myText"] = Label()` ohne Parameter initialisieren! sonst kannst Du die `self["myText"].setText(self.text)` in Zeile 39 nicht ausführen!

zu Zeile 36:

`self.onShown.append(self.setMyText)` damit was von Anfang an drinsteht!

Ergebnis:



Fortsetzung src zu: AutoSleepScreen

```
51     def setSleep(self):
52         self.changed = True
53         if config.plugins.AutoSleep.enable.value:
54             config.plugins.AutoSleep.enable.setValue(False)
55         else:
56             config.plugins.AutoSleep.enable.setValue(True)
57         self.updateSettings()
58
59     def exit(self):
60         if self.changed:
61             config.plugins.AutoSleep.enable.save()
62         self.close(None)
63
64 #####
65
66 def main(session, **kwargs):
67     session.open(AutoSleepScreen)
68
69 #####
70 # start and stop enigma2 & and watch output in telnet
71
72 def autostart(reason, **kwargs):
73     print blank, line
74     if reason == 0:
75         print "[AutoSleep] - autostart sleep enabled: ",
76         config.plugins.AutoSleep.enable.getValue()
77     else:
78         print "[AutoSleep] - autostop sleep enabled: ",
79         config.plugins.AutoSleep.enable.getValue()
80     print tut_vers
81     print line, blank
82
83     if config.plugins.AutoSleep.enable.value:
84         time.sleep(10)
85
86 #####
87 def Plugins(**kwargs):
88     return [
89         PluginDescriptor(
90             where = PluginDescriptor.WHERE_AUTOSTART,
91             fnc = autostart),
92         PluginDescriptor(
93             name = "10 Set Auto Sleep",
94             description = "lesson 10 - Ihad.tv e2-tutorial",
95             where = PluginDescriptor.WHERE_PLUGINMENU,
96             icon = "../ihad_tut.png",
97             fnc = main)]
98
```

Erklärung AutoSleepScreen Klasse:

Das sollte für dich jetzt kein Problem mehr sein. Neu sind hier die Configs und der Autostart PluginDescriptor.

zu Zeile 10:

import von config, ConfigSubsection, ConfigYesNo

zu Zeile 14:

config.plugins.AutoSleep = ConfigSubsection() **erzeugen von Enigma Zusatz Konfigurationen**

zu Zeile 15:

config.plugins.AutoSleep.enable = ConfigYesNo(default = False)
eine variable die 0/1 speichern kann ergänzen

zu Zeile 73-83:

die Funktion autostart wird vom PluginDescriptor in Zeile 91 aufgerufen und überprüft ob **config.plugins.AutoSleep.enable** gesetzt ist oder nicht. wenn es gesetzt ist weird der Bootvorgang deiner dreambox für 10 sec. angehalten. Beobachten kannst du das ganze im Telnet, wenn du die Box startest oder stopst.

Ergebnis:



"11 Start other plugin"

Screen zum Starten des Bildbetrachte

```
1 # Ihad.tv enigma2-plugin tutorial 2010
2 # lesson 11
3 # by emanuel
4 #####
5
6 from Screens.Screen import Screen
7 from Screens.MessageBox import MessageBox
8 from Components.Label import Label
9 from Components.ActionMap import ActionMap
10 from Plugins.Plugin import PluginDescriptor
11 from Tools.Directories import fileExists
12
13 #####
14
15 class MyPluginStartScreen(Screen):
16     skin = """
17         <screen position="130,150" size="460,150" title="Ihad.tv e2-tutorial
18 lesson 11" >
19             <widget name="myText" position="10,20" size="400,50"
20 valign="center" halign="center" zPosition="2" foregroundColor="white"
21 font="Regular;22"/>
22             <widget name="myRedBtn" position="10,110" size="100,40"
23 backgroundColor="red" valign="center" halign="center" zPosition="2"
24 foregroundColor="white" font="Regular;20"/>
25             <widget name="myGreenBtn" position="120,110" size="100,40"
26 backgroundColor="green" valign="center" halign="center" zPosition="2"
27 foregroundColor="white" font="Regular;20"/>
28         </screen>"""
29
30     def __init__(self, session, args = 0):
31         self.session = session
32         Screen.__init__(self, session)
33
34         self["myText"] = Label("Press green or ok button to start\nPicture
35 Player plugin!")
36         self["myRedBtn"] = Label(_("Cancel"))
37         self["myGreenBtn"] = Label(_("OK"))
38         self["myActionsMap"] = ActionMap(["SetupActions", "ColorActions"],
39 {
40     "ok": self.startPicplayer,
41     "green": self.startPicplayer,
42     "red": self.close,
43     "cancel": self.close,
44 }, -1)
45
46     def startPicplayer(self):
47         if
48 fileExists("/usr/lib/enigma2/python/Plugins/Extensions/PicturePlayer/plugin.py"):
49             from Plugins.Extensions.PicturePlayer.plugin import *
50             self.session.open(picshow)
51         else:
52             self.session.open(MessageBox, "No Picture Player found!",
53 MessageBox.TYPE_ERROR)
54
55 #####
56
57
```

Fortsetzung src zu: MyPluginStartScreen

```
50 def main(session, **kwargs):
51     session.open(MyPluginStartScreen)
52
53 #####
54
55 def Plugins(**kwargs):
56     return PluginDescriptor(
57         name="11 Start other plugin",
58         description="lesson 11 - Ihad.tv e2-tutorial",
59         where = PluginDescriptor.WHERE_PLUGINMENU,
60         icon="../ihad_tut.png",
61         fnc=main)
62
```

Erklärung MyPluginStartScreen Klasse:

Das sollte für dich jetzt kein Problem mehr sein. Screen wurden ja schon öfters importiert.

zu Zeile 10/42:

import von fileExists, alles vom PicturePlayerplugin

Ergebnis:

