tianocore

UEFI & EDK II Training

How to Write a UEFI Application – Linux Lab

tianocore.org

See also LabGuide.md for Copy & Paste examples in labs

tianocore

Lesson Objective

- UEFI Application with PCDs
- Simple UEFI Application
- Add functionality to UEFI Application
- Using EADK with UEFI Application

UEFI APPLICATION W/ PCDS

tianocore

EDK II PCD's Purpose and Goals

Review

Documentaton : MdeModulePkg/Universal/PCD/Dxe/Pcd.inf

Purpose

- Establishes platform common definitions
- Build-time/Run-time aspects
- Binary Editing Capabilities

Goals

- Simplify porting
- Easy to associate with a module or platform

PCD Syntax

Review

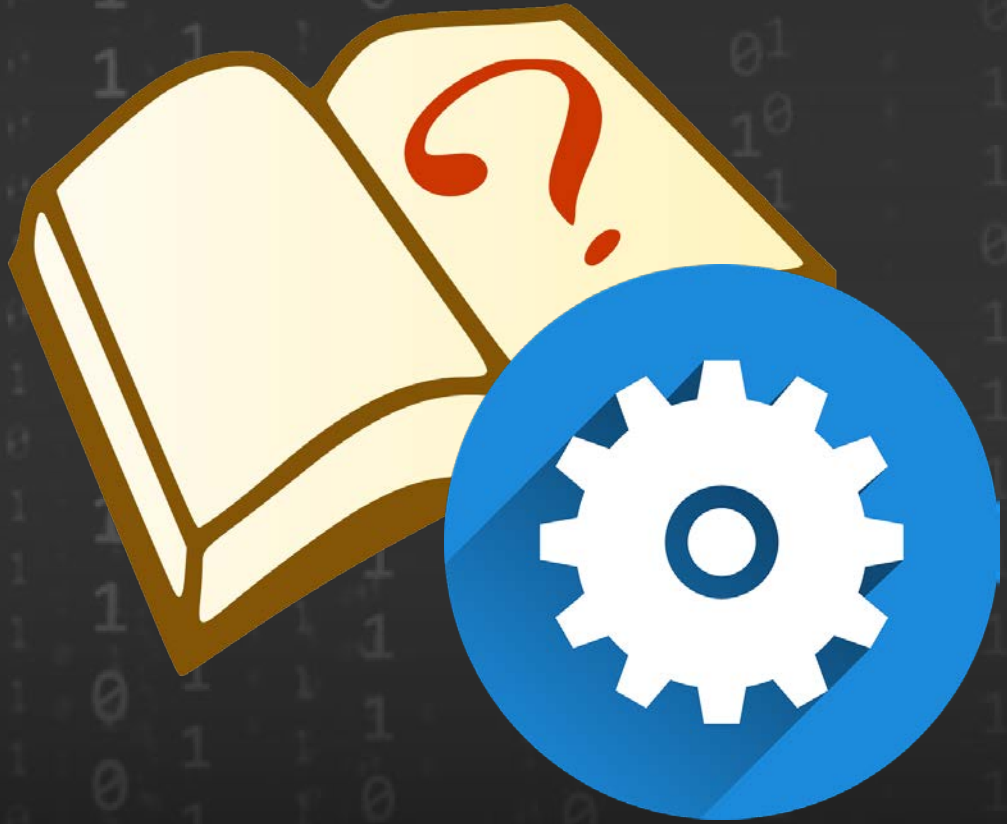
PCDs can be located anywhere within the Workspace even though a different package will use those PCDs for a given project

.DEC	.INF	.DSC
Define PCD	Reference PCD	Modify PCD
Package	Module	Platform

tianocore

Lab 1: Writing UEFI Applications with PCDs

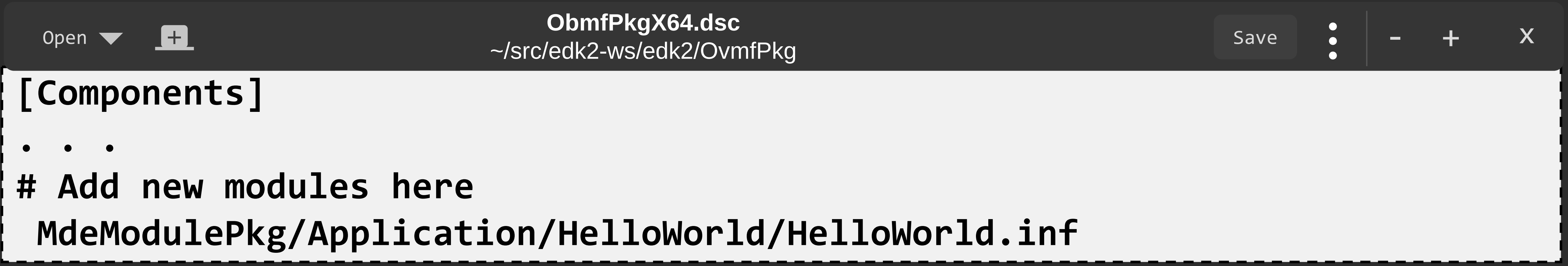
In this lab, you’ll learn how to write UEFI applications with PCDs.



EDK II HelloWorld App Lab

First Setup for Building EDK II for OVMF, See Lab Setup

Edit and add the following line (at the end of the file)
Edit OvmfPkg/OvmfPkgX64.dsc add HelloWorld.inf - Save



```
[Components]
. . .
# Add new modules here
 MdeModulePkg/Application/HelloWorld/HelloWorld.inf
```

Build the OvmfPkgX64 from Terminal Prompt (Cnt-Alt-T)

```
 bash$ cd ~/src/edk2-ws/edk2
 bash$ build  -D ADD_SHELL_STRING
```

tianocore

EDK II HelloWorld App Lab

1. Copy the HelloWorld.efi to the ~run-ovmf/hda-contents directory

```
   bash$ cd ~/run-ovmf/hda-contents
   bash$ cp ~/src/edk2-ws/Build/OvmfX64/DEBUG_GCC5/X64/HelloWorld.efi
```

2. CD to the run-ovmf directory and run Qemu with the RunQemu.sh shell

```
   bash$ cd ~/run-ovmf
   bash$ . RunQemu.sh
```

3. At the UEFI Shell prompt

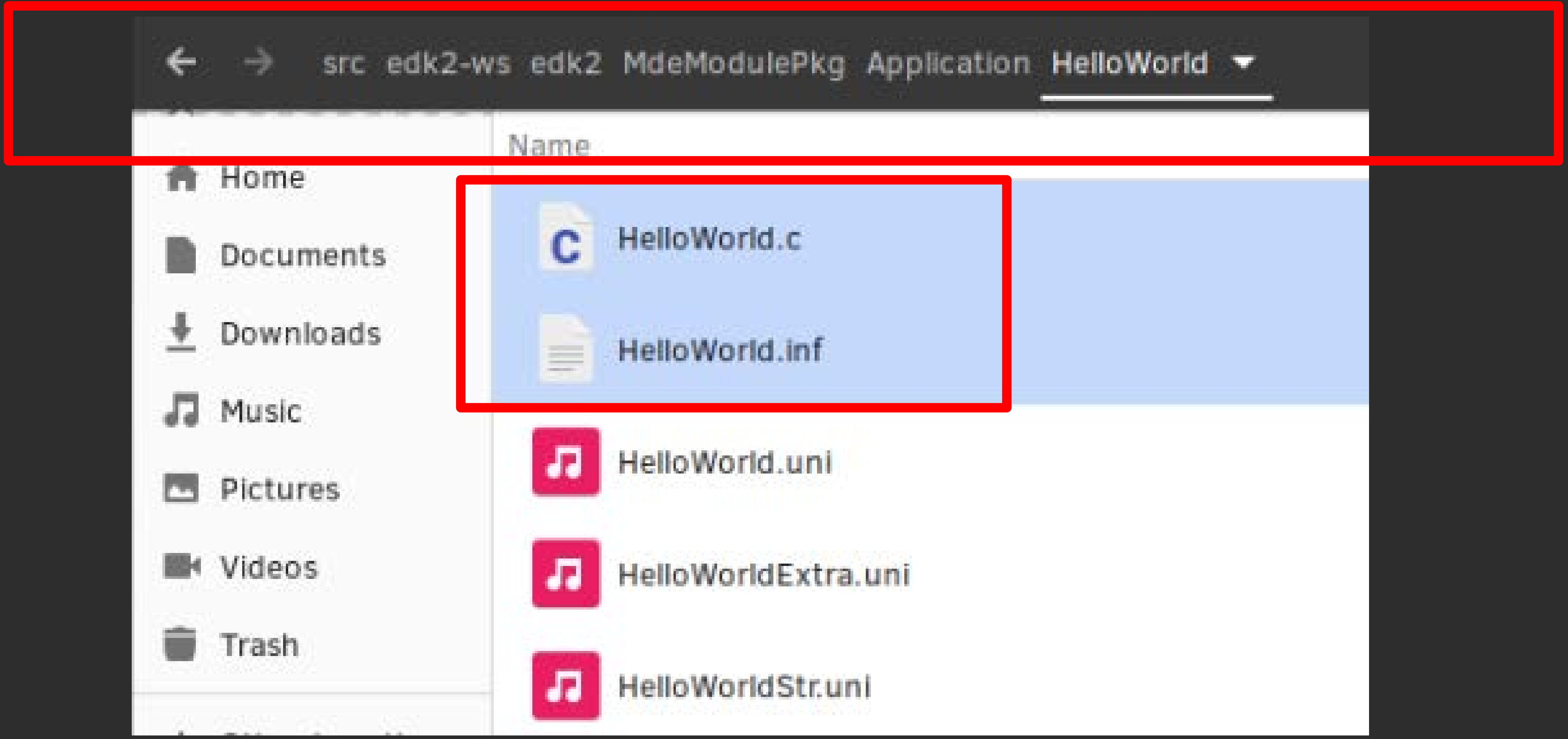
```
Shell> Helloworld
UEFI Hello World!
Shell>
```

How can we force the HelloWorld application to print out 3 times ?

tianocore

EDK II HelloWorld App Lab

MdeModulePkg/Application/HelloWorld



HelloWorld.c
HelloWorld.inf
HelloWorld.uni
HelloWorldExtra.uni
HelloWorldStr.uni

EDK II HelloWorld App

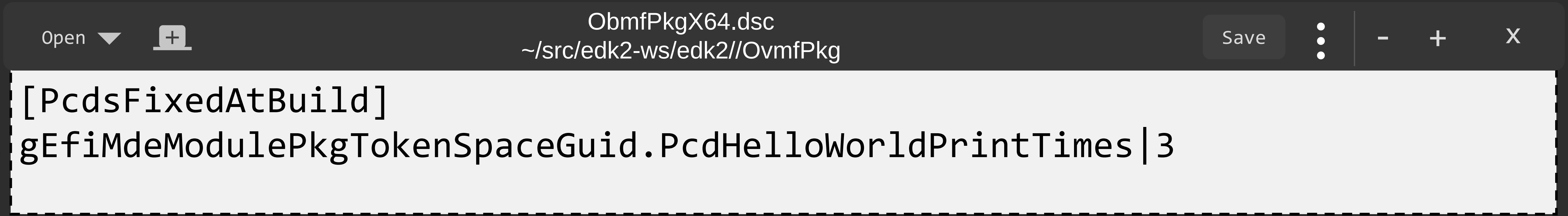
Source HelloWorld.c

```
EFI_STATUS
EFIAPI
UefiMain (
  IN EFI_HANDLE        ImageHandle,
  IN EFI_SYSTEM_TABLE  *SystemTable
  )
{
  UINT32 Index;
  Index = 0;
  // Three PCD type (FeatureFlag, UINT32
  // and String) are used as the sample.
  if (FeaturePcdGet (PcdHelloWorldPrintEnable)) {
  for (Index = 0; Index < PcdGet32   (PcdHelloWorldPrintTimes); Index ++) {

    // Use UefiLib Print API to print
      // string to UEFI console

        Print ((CHAR16*)PcdGetPtr (PcdHelloWorldPrintString));
    }
  }

  return EFI_SUCCESS;
}
```

EDK II HelloWorld App Solution

1. Edit the file OvmfPkg/OvmfPkgX64.dsc

After the section **`[PcdsFixedAtBuild]`** (search for “**`PcdsFixedAtBuild`**” or “**`Hello`**”)



```
[PcdsFixedAtBuild]
gEfiMdeModulePkgTokenSpaceGuid.PcdHelloWorldPrintTimes|3
```

2. Re-Build – Cd to ~/src/edk2-ws/edk2~ dir

```
bash$ build -D ADD_SHELL_STRING
```

3. Copy Helloworld.efi

```
bash$ cd ~/run-ovmf/hda-contents
bash$ cp ~/src/Build/OvmfX64/DEBUG_GCC5/X64/HelloWorld.efi .
```

tianocore

EDK II HelloWorld App Solution

4. Run Qemu

```
bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```

5. At the Shell prompt

```
Shell> Helloworld
UEFI Hello World!
UEFI Hello World!
UEFI Hello World!
Shell>
```

Exit QEMU

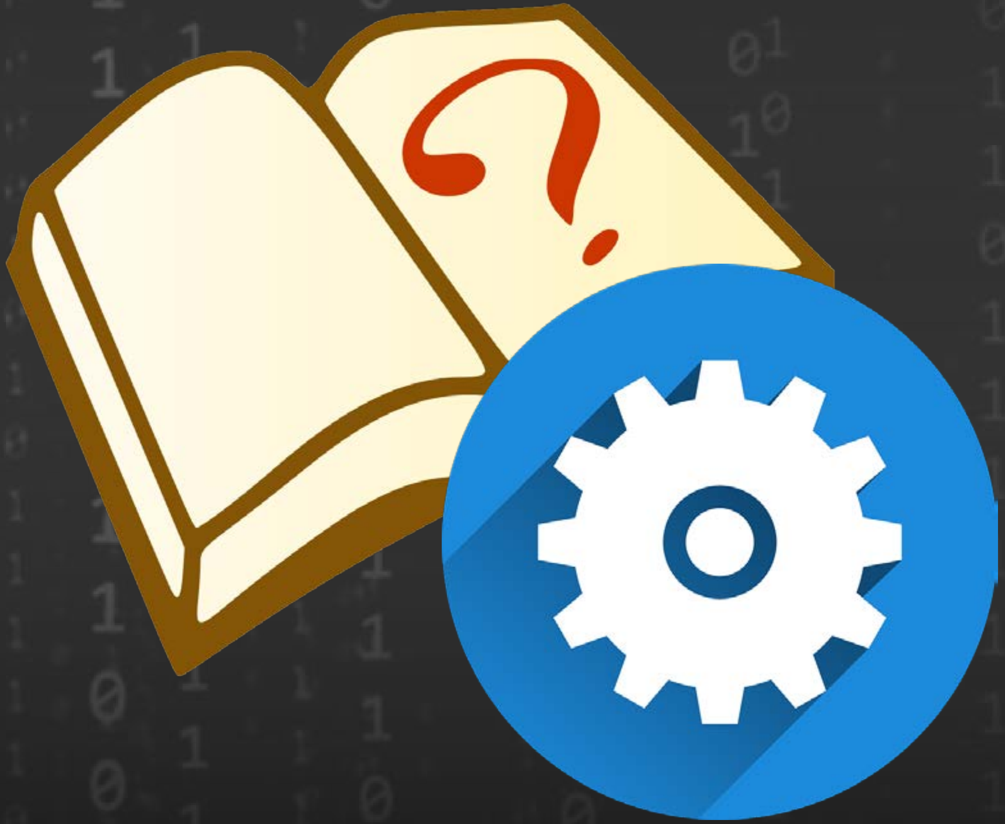
How can we change the **string** of the HelloWorld application?

Also see ~src/edk2-ws/edk2/MdeModulePkg/MdeModulePkg.Dec

tianocore

Lab 2: Write a Simple UEFI Applications

In this lab, you’ll learn how to write simple UEFI applications.



LAB 2 Writing a Simple UEFI Application

In this lab, you'll learn how to write simple UEFI applications.

"C" file

```
EFI_STATUS
EFIAPI
UefiMain (
  IN EFI_HANDLE           ImageHandle,
  IN EFI_SYSTEM_TABLE   *SystemTable
  )
{
  return EFI_SUCCESS;
}
```

.inf file

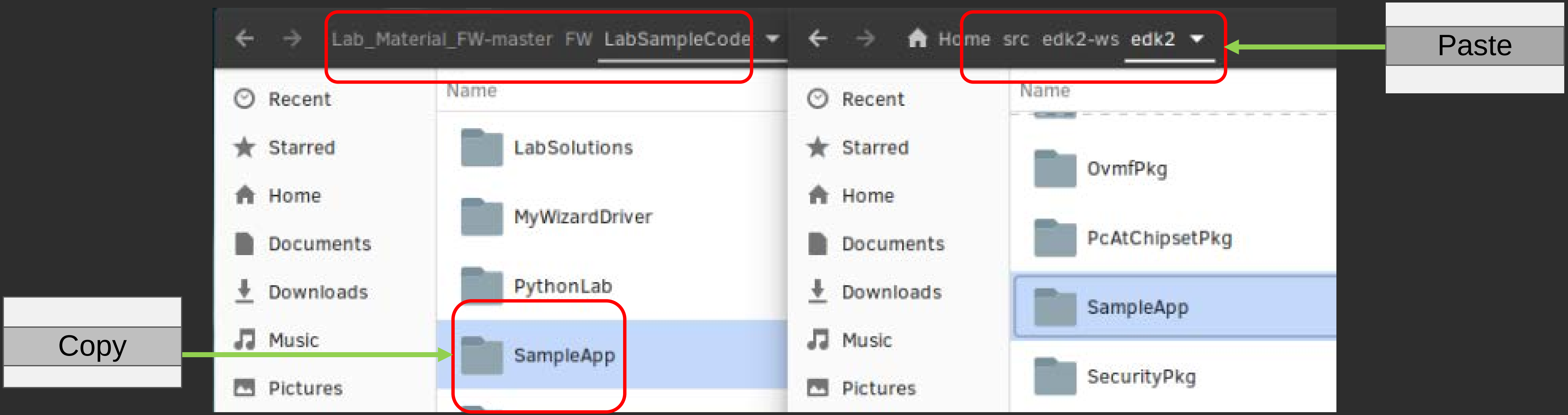
```
[Defines]
  INF_VERSION          =
  BASE_NAME            =
  FILE_GUID            =
  MODULE_TYPE          =
  VERSION_STRING       =
  ENTRY_POINT          =

[Sources]

[Packages]

[LibraryClasses]
```

- What goes into a Simplest "C"
- Start with what should go into the Simplest .INF file

tianocore

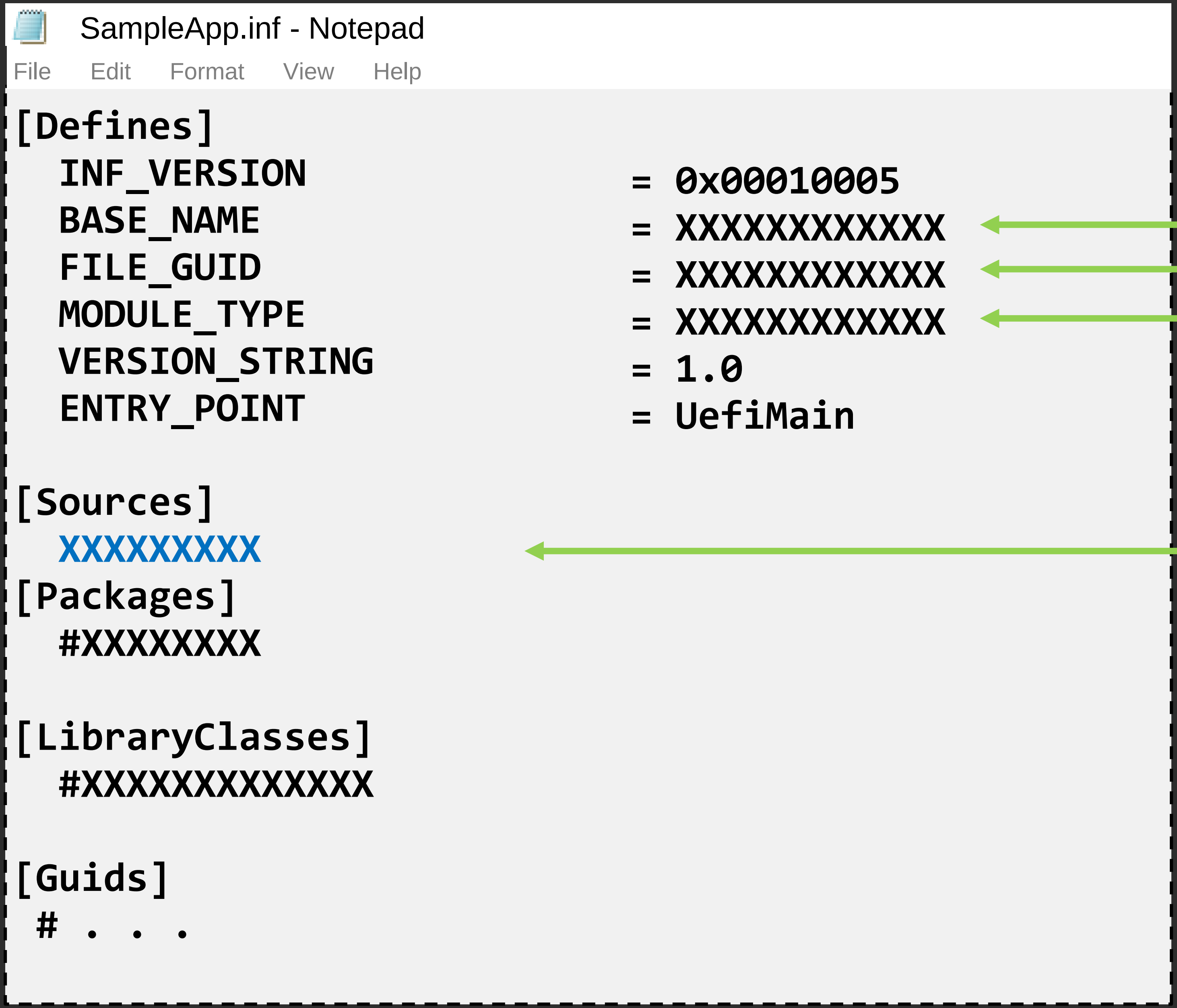
Application Lab –start with .c and .inf template

1. Copy the `LabSampleCode/SampleApp` directory to `~/src/edk2-ws/edk2`



2. Edit `SampleApp.inf`
   - Look in the INF for **“XXXXXXXXXXX”** sections that will need information
   - Create Name & GUID, and then fill in the MODULE_TYPE

Lab 2: Sample Application INF file



```
[Defines]
  INF_VERSION                   = 0x00010005
  BASE_NAME                     = XXXXXXXXXXXX
  FILE_GUID                     = XXXXXXXXXXXX
  MODULE_TYPE                   = XXXXXXXXXXXX
  VERSION_STRING                = 1.0
  ENTRY_POINT                   = UefiMain

[Sources]
  XXXXXXXXX
[Packages]
  #XXXXXXXX

[LibraryClasses]
  #XXXXXXXXXXXXX

[Guids]
 # . . .
```

SampleApp

Get a GUID

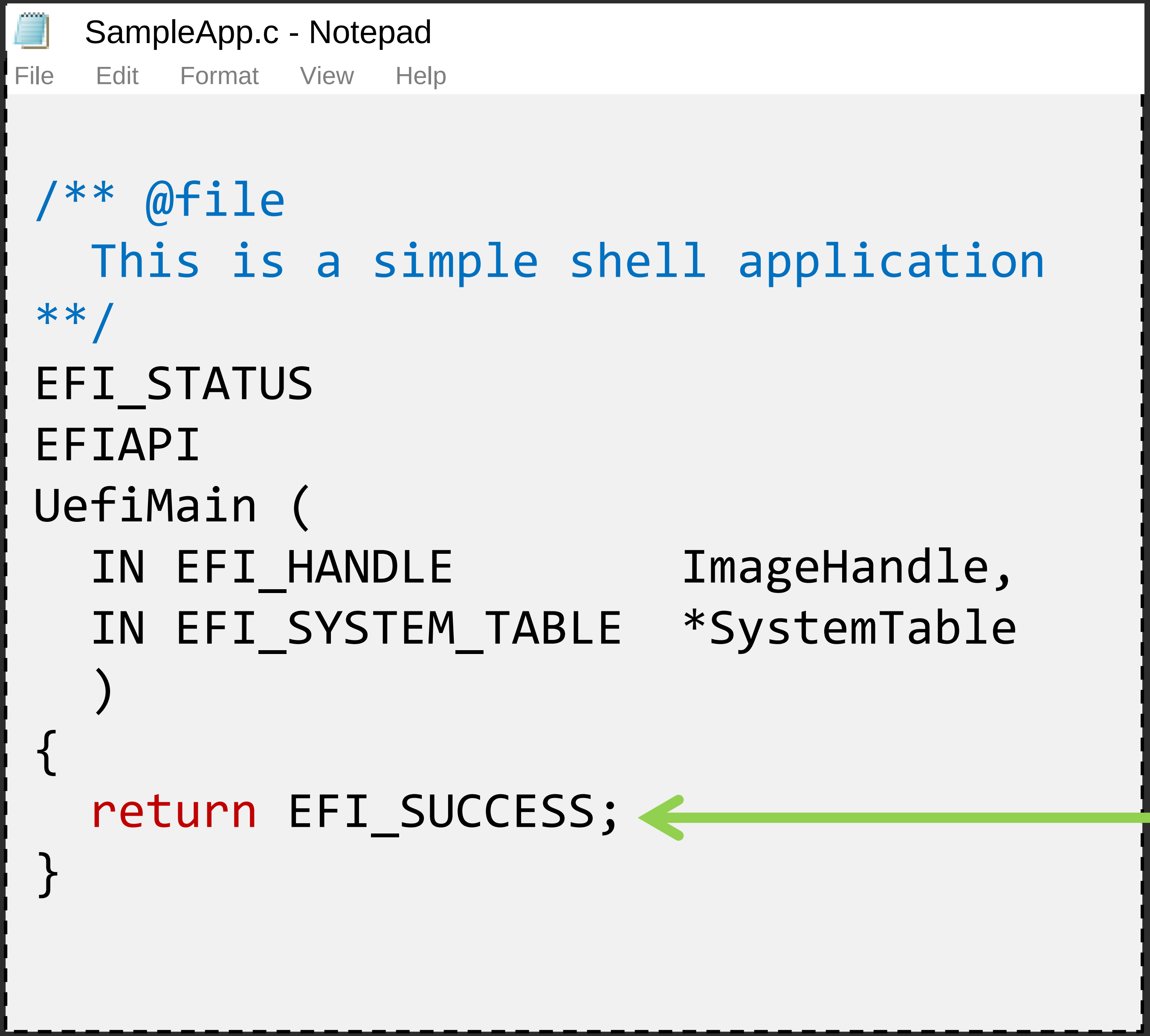
UEFI_APPLICATION

SampleApp.c

Get a GUID guidgenerator.com/

Copy and paste LabGuide.md

Lab 2: Sample Application ‘C’ file



```
/** @file
  This is a simple shell application
**/
EFI_STATUS
EFIAPI
UefiMain (
  IN EFI_HANDLE        ImageHandle,
  IN EFI_SYSTEM_TABLE  *SystemTable
  )
{
  return EFI_SUCCESS;
}
```

Does not do anything but return Success

Lab 2: Will it compile now?

Not yet …

1. Need to add headers to the .C file
2. Need to add a reference to INF from the platform DSC
3. Need to add a few Package dependencies and libraries to the .INF

tianocore

Application Lab – Update Files

1. .DSC (`OvmfPkg/OvmfPkgX64.dsc`)

`[Components . . .]`

Add INF to components section, before build options
Hint: add after comment: `# Add new modules here`

```
SampleApp/SampleApp.inf
```

2. .INF File (**`SampleApp/SampleApp.inf`**)

Packages (all depend on MdePkg)

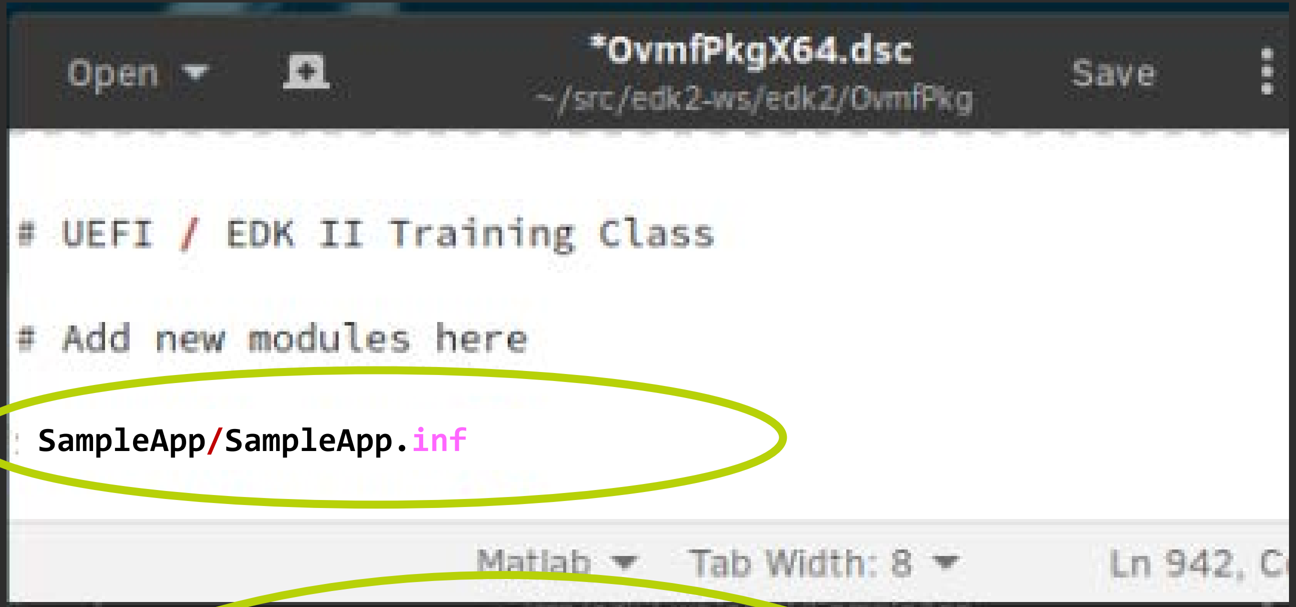
```
[Packages]
    MdePkg/MdePkg.dec
[LibraryClasses]
    UefiApplicationEntryPoint
```

3. .C file - Header references File (**`SampleApp/SampleApp.c`**)

```
 #include <Uefi.h>
#include <Library/UefiApplicationEntryPoint.h>
```

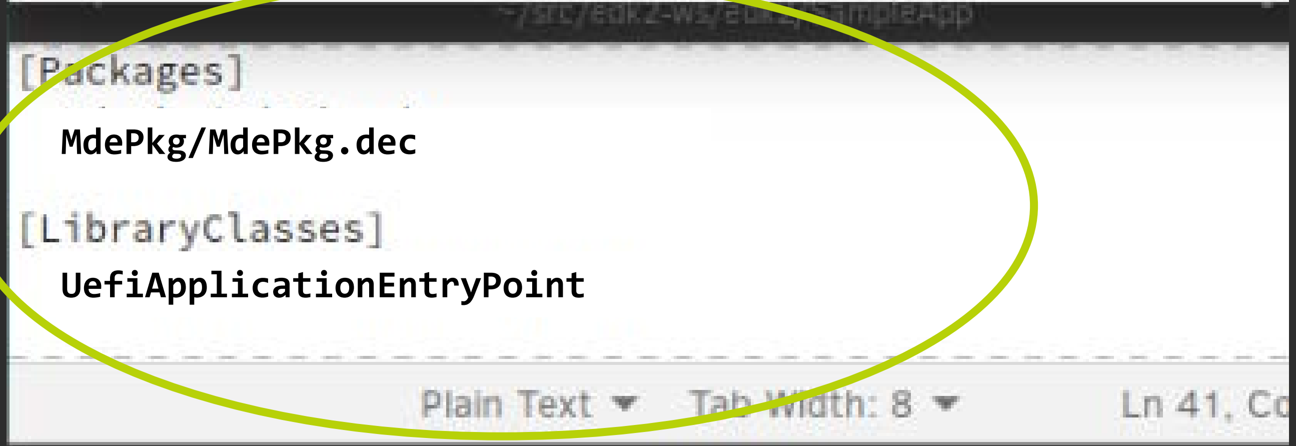
Copy and paste LabGuide.md

Lab 2: Lab cont. Solution



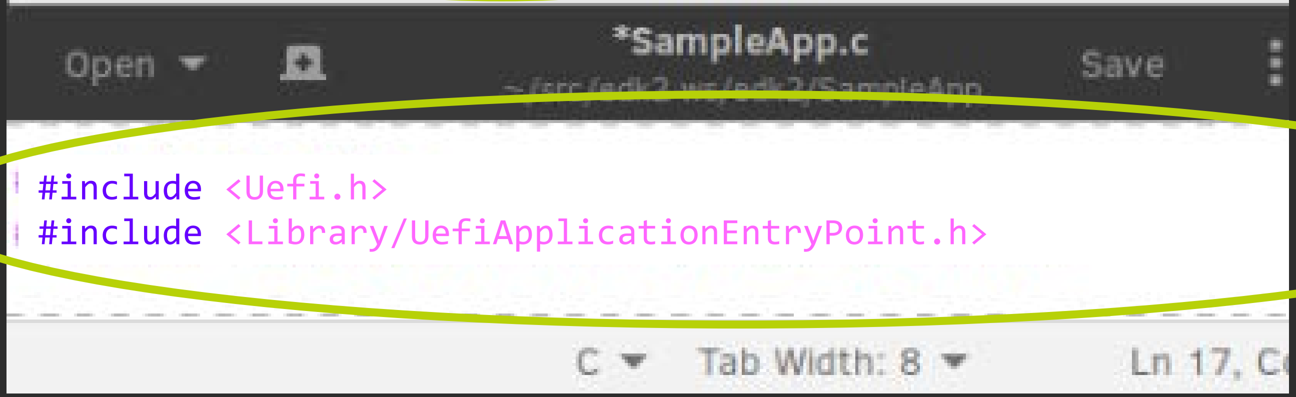
```
# UEFI / EDK II Training Class
# Add new modules here
SampleApp/SampleApp.inf
```

OvmfPkg/OvmfPkgX64.dsc



```
[Packages]
  MdePkg/MdePkg.dec
[LibraryClasses]
  UefiApplicationEntryPoint
```

SampleApp/SampleApp.inf



```
#include <Uefi.h>
#include <Library/UefiApplicationEntryPoint.h>
```

SampleApp/SampleApp.c

tianocore

Lab 2 : Will it compile now?

Yes, Build SampleApp – Cd to ~/src/edk2-ws/edk2 directory

```
bash$ build –D ADD_SHELL_STRING
```

Copy SampleApp.efi to hda-contents

```
 bash$ cd ~/run-ovmf/hda-contents
 bash$ cp ~/src/edk2-ws/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp.efi .
```

Test by Invoking Qemu

```
 bash$ cd ~/run-ovmf
 bash$ . RunQemu.sh
```

Run the application from the shell

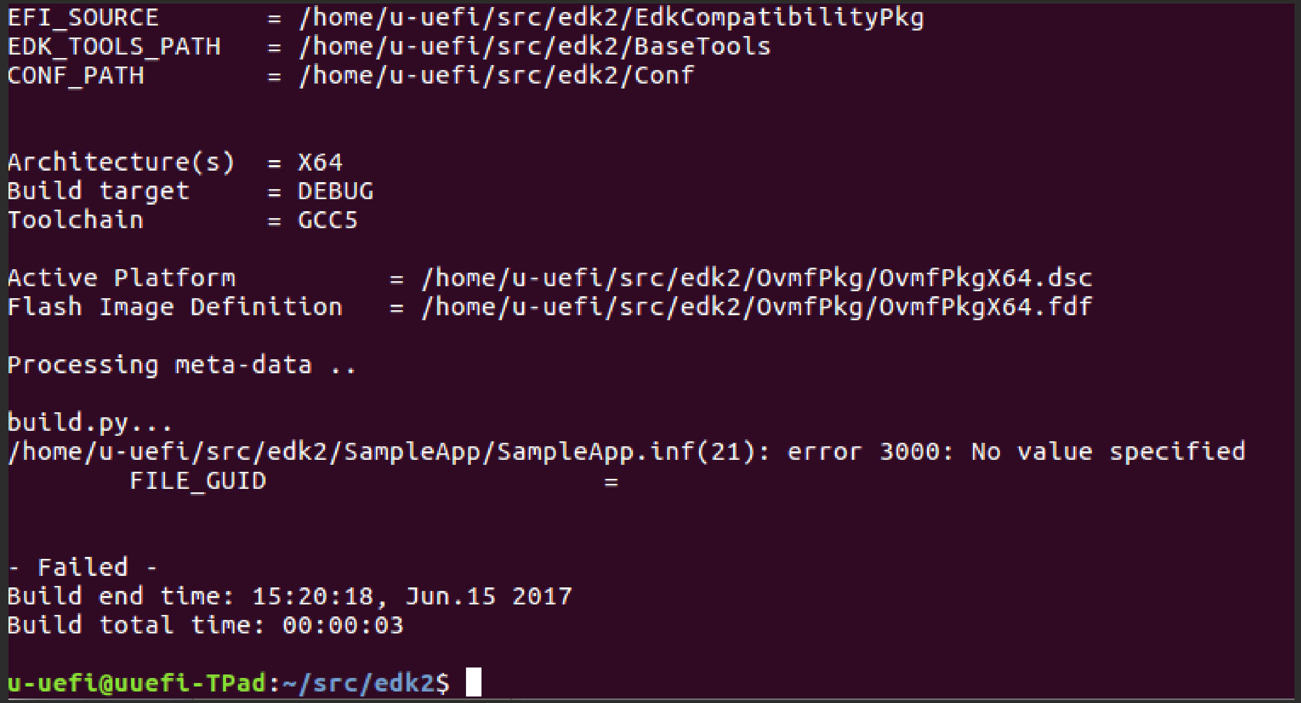
```
    Shell> SampleApp
    Shell>
```

Notice that the program will immediately unload because the main function is empty

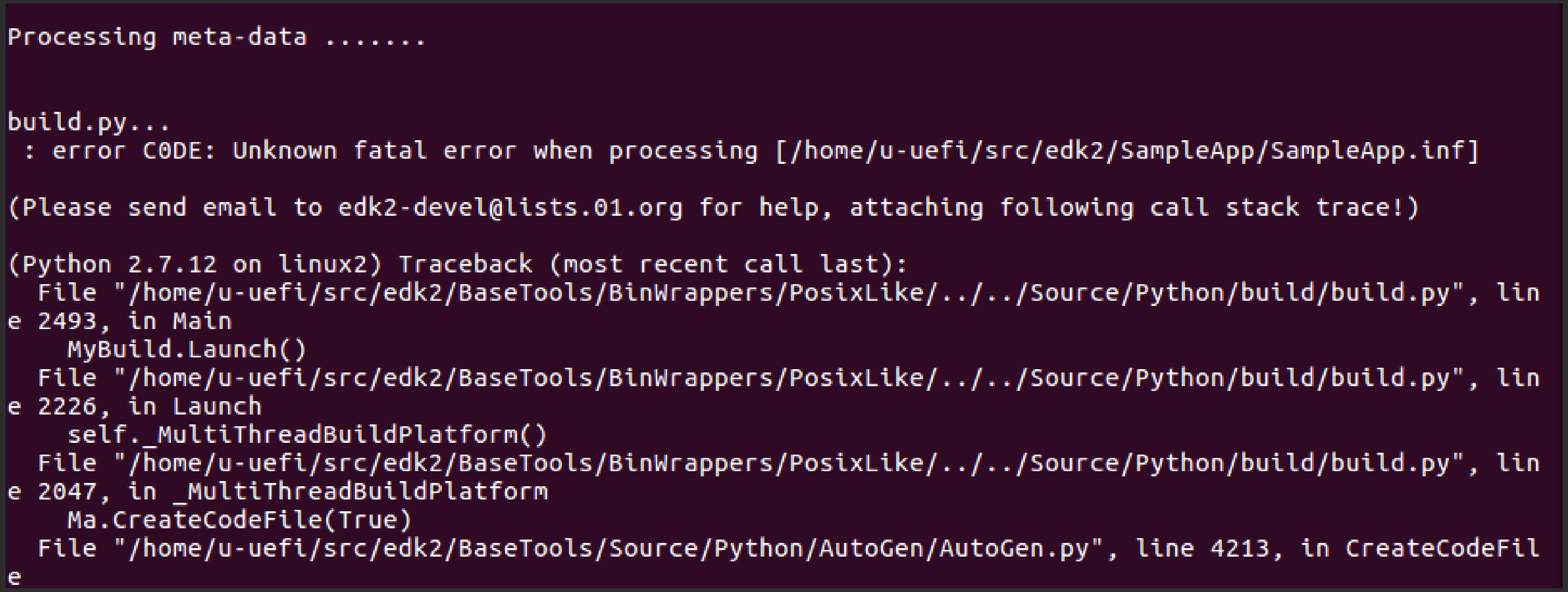
Exit QEMU

Possible Build Errors

Error on SampleApp.inf



```
EFI_SOURCE        = /home/u-uefi/src/edk2/EdkCompatibilityPkg
EDK_TOOLS_PATH    = /home/u-uefi/src/edk2/BaseTools
CONF_PATH         = /home/u-uefi/src/edk2/Conf


Architecture(s)   = X64
Build target      = DEBUG
Toolchain         = GCC5

Active Platform          = /home/u-uefi/src/edk2/OvmfPkg/OvmfPkgX64.dsc
Flash Image Definition   = /home/u-uefi/src/edk2/OvmfPkg/OvmfPkgX64.fdf

Processing meta-data ..

build.py...
/home/u-uefi/src/edk2/SampleApp/SampleApp.inf(21): error 3000: No value specified
        FILE_GUID                      =


- Failed -
Build end time: 15:20:18, Jun.15 2017
Build total time: 00:00:03

u-uefi@uuefi-TPad:~/src/edk2$
```

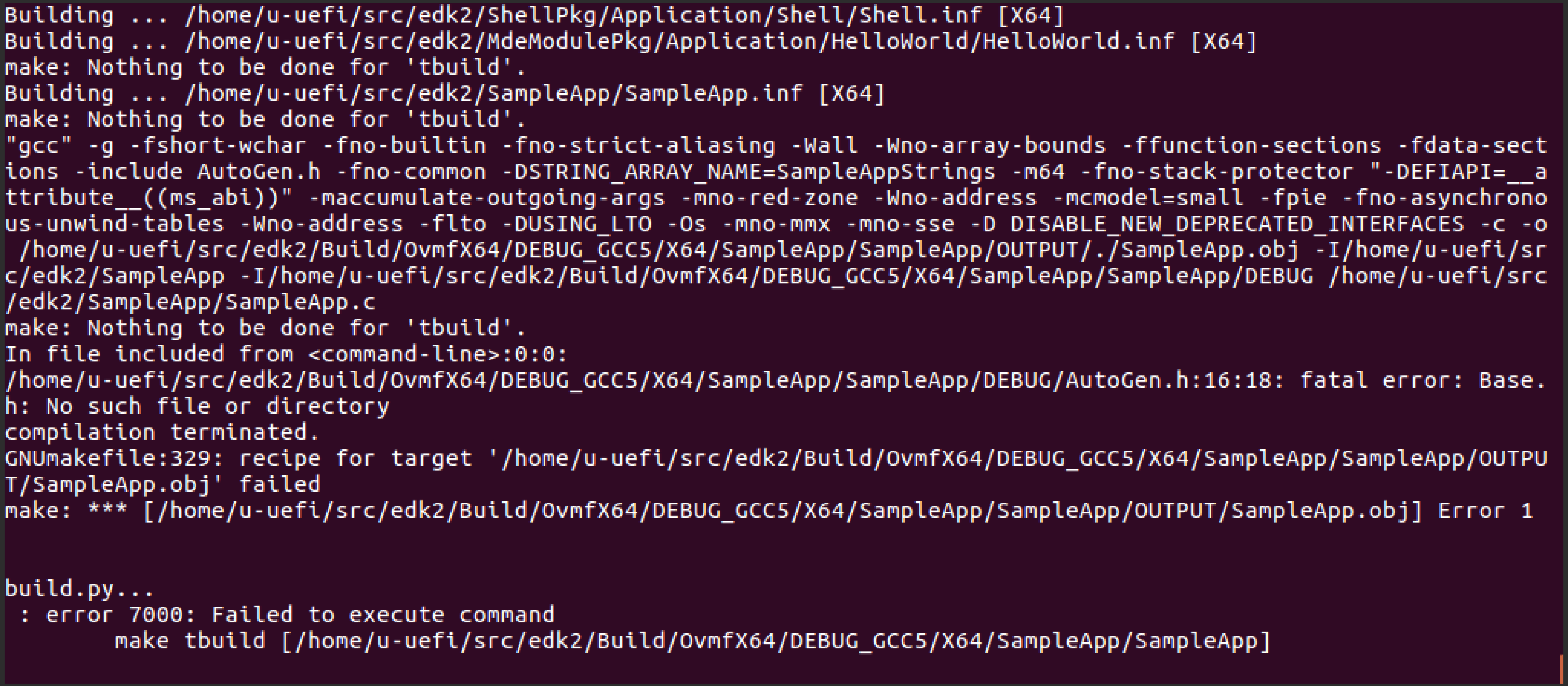


```
Processing meta-data .......


build.py...
 : error C0DE: Unknown fatal error when processing [/home/u-uefi/src/edk2/SampleApp/SampleApp.inf]

(Please send email to edk2-devel@lists.01.org for help, attaching following call stack trace!)

(Python 2.7.12 on linux2) Traceback (most recent call last):
  File "/home/u-uefi/src/edk2/BaseTools/BinWrappers/PosixLike/../../Source/Python/build/build.py", lin
e 2493, in Main
    MyBuild.Launch()
  File "/home/u-uefi/src/edk2/BaseTools/BinWrappers/PosixLike/../../Source/Python/build/build.py", lin
e 2226, in Launch
    self._MultiThreadBuildPlatform()
  File "/home/u-uefi/src/edk2/BaseTools/BinWrappers/PosixLike/../../Source/Python/build/build.py", lin
e 2047, in _MultiThreadBuildPlatform
    Ma.CreateCodeFile(True)
  File "/home/u-uefi/src/edk2/BaseTools/Source/Python/AutoGen/AutoGen.py", line 4213, in CreateCodeFil
e
```

The FILE_GUID was invalid or not updated from “XXX…” to a proper formatted GUID

Possible Build Errors

Error on SampleApp.inf



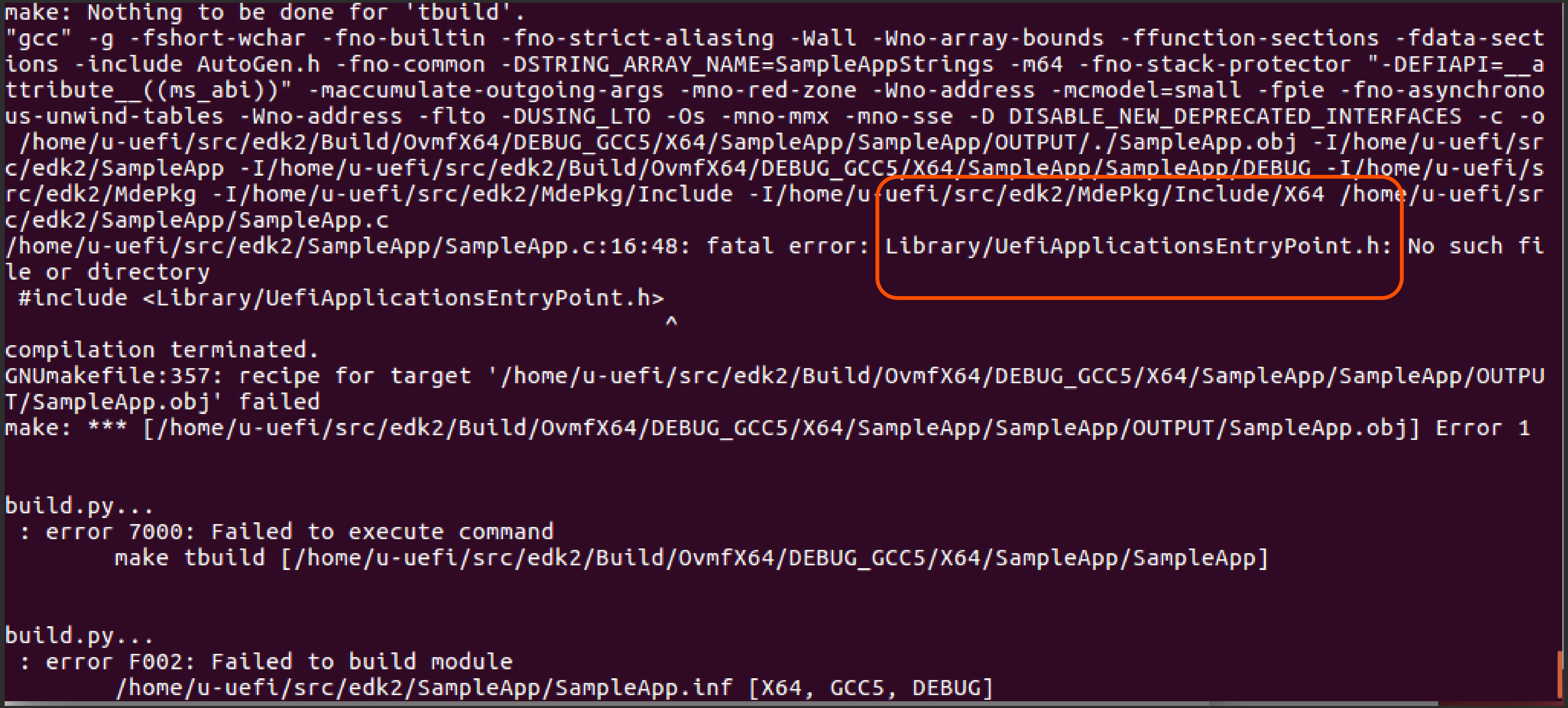
```
Building ... /home/u-uefi/src/edk2/ShellPkg/Application/Shell/Shell.inf [X64]
Building ... /home/u-uefi/src/edk2/MdeModulePkg/Application/HelloWorld/HelloWorld.inf [X64]
make: Nothing to be done for 'tbuild'.
Building ... /home/u-uefi/src/edk2/SampleApp/SampleApp.inf [X64]
make: Nothing to be done for 'tbuild'.
"gcc" -g -fshort-wchar -fno-builtin -fno-strict-aliasing -Wall -Wno-array-bounds -ffunction-sections -fdata-sect
ions -include AutoGen.h -fno-common -DSTRING_ARRAY_NAME=SampleAppStrings -m64 -fno-stack-protector "-DEFIAPI=__a
ttribute__((ms_abi))" -maccumulate-outgoing-args -mno-red-zone -Wno-address -mcmodel=small -fpie -fno-asynchrono
us-unwind-tables -Wno-address -flto -DUSING_LTO -Os -mno-mmx -mno-sse -D DISABLE_NEW_DEPRECATED_INTERFACES -c -o
 /home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/OUTPUT/./SampleApp.obj -I/home/u-uefi/sr
c/edk2/SampleApp -I/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG /home/u-uefi/src
/edk2/SampleApp/SampleApp.c
make: Nothing to be done for 'tbuild'.
In file included from <command-line>:0:0:
/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG/AutoGen.h:16:18: fatal error: Base.
h: No such file or directory
compilation terminated.
GNUmakefile:329: recipe for target '/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/OUTPU
T/SampleApp.obj' failed
make: *** [/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/OUTPUT/SampleApp.obj] Error 1


build.py...
 : error 7000: Failed to execute command
        make tbuild [/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp]
```

The [Packages] was invalid or did not specify MdePkg/MdePkg.dec properly

tianocore

Possible Build Errors

GCC compiler Error on SampleApp.c



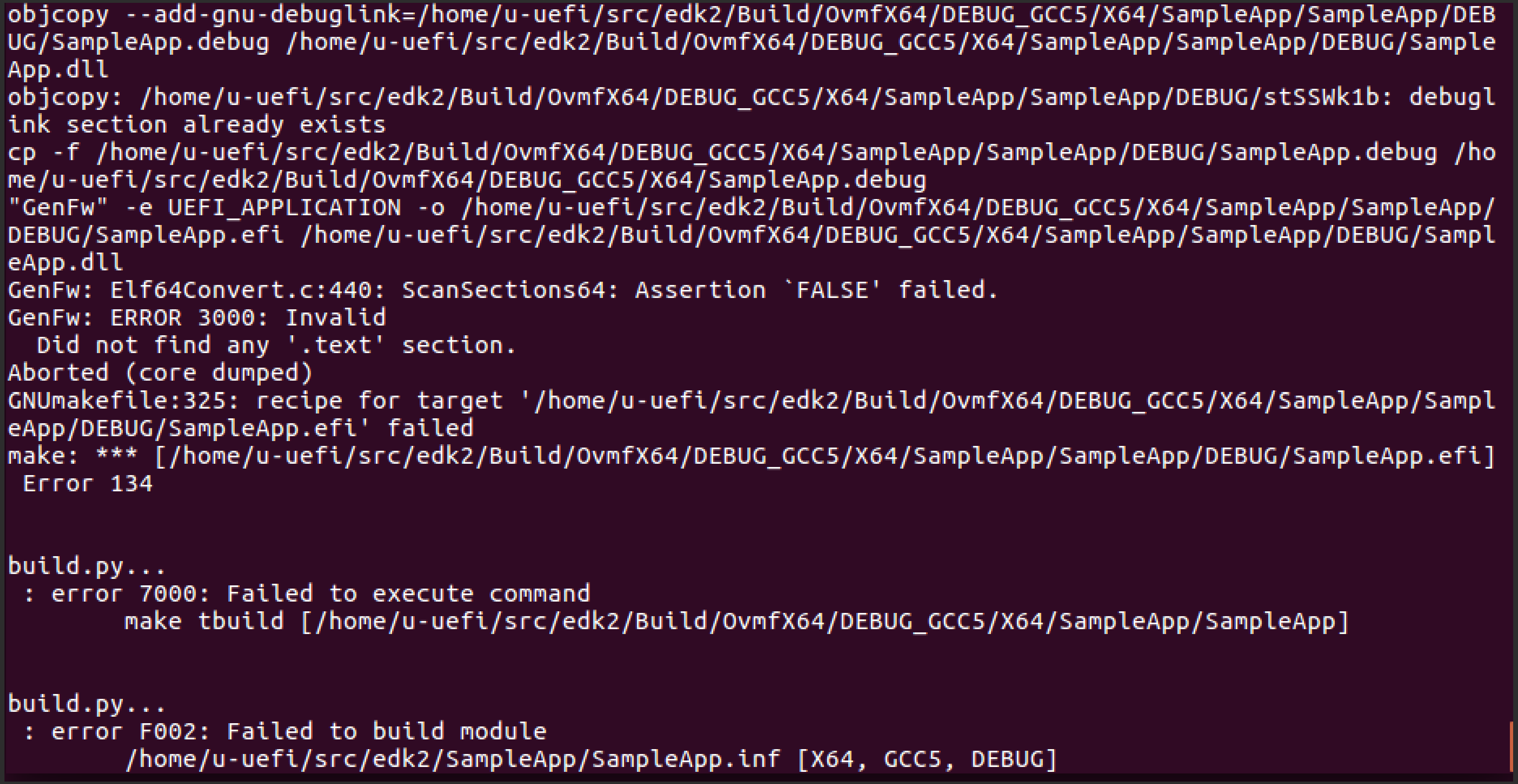
```
make: Nothing to be done for 'tbuild'.
"gcc" -g -fshort-wchar -fno-builtin -fno-strict-aliasing -Wall -Wno-array-bounds -ffunction-sections -fdata-sect
ions -include AutoGen.h -fno-common -DSTRING_ARRAY_NAME=SampleAppStrings -m64 -fno-stack-protector "-DEFIAPI=__a
ttribute__((ms_abi))" -maccumulate-outgoing-args -mno-red-zone -Wno-address -mcmodel=small -fpie -fno-asynchrono
us-unwind-tables -Wno-address -flto -DUSING_LTO -Os -mno-mmx -mno-sse -D DISABLE_NEW_DEPRECATED_INTERFACES -c -o
 /home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/OUTPUT/./SampleApp.obj -I/home/u-uefi/sr
c/edk2/SampleApp -I/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG -I/home/u-uefi/s
rc/edk2/MdePkg -I/home/u-uefi/src/edk2/MdePkg/Include -I/home/u-uefi/src/edk2/MdePkg/Include/X64 /home/u-uefi/sr
c/edk2/SampleApp/SampleApp.c
/home/u-uefi/src/edk2/SampleApp/SampleApp.c:16:48: fatal error: Library/UefiApplicationsEntryPoint.h: No such fi
le or directory
 #include <Library/UefiApplicationsEntryPoint.h>
                                                ^
compilation terminated.
GNUmakefile:357: recipe for target '/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/OUTPU
T/SampleApp.obj' failed
make: *** [/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/OUTPUT/SampleApp.obj] Error 1


build.py...
 : error 7000: Failed to execute command
        make tbuild [/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp]


build.py...
 : error F002: Failed to build module
        /home/u-uefi/src/edk2/SampleApp/SampleApp.inf [X64, GCC5, DEBUG]
```

The #include <Library/UefiApplicationEntryPoint.h> has a typo (“Application” not “Applications”)

tianocore

Possible Build Errors

GCC compiler Error on SampleApp.c



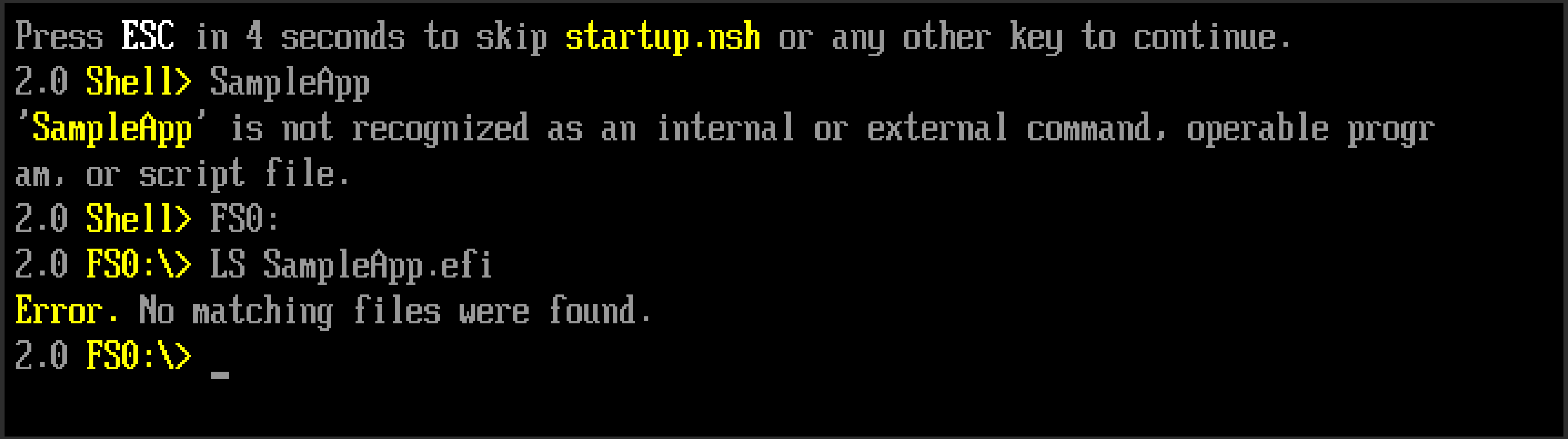
```
objcopy --add-gnu-debuglink=/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEB
UG/SampleApp.debug /home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG/Sample
App.dll
objcopy: /home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG/stSSWk1b: debugl
ink section already exists
cp -f /home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG/SampleApp.debug /ho
me/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp.debug
"GenFw" -e UEFI_APPLICATION -o /home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/
DEBUG/SampleApp.efi /home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG/Sampl
eApp.dll
GenFw: Elf64Convert.c:440: ScanSections64: Assertion `FALSE' failed.
GenFw: ERROR 3000: Invalid
  Did not find any '.text' section.
Aborted (core dumped)
GNUmakefile:325: recipe for target '/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/Sampl
eApp/DEBUG/SampleApp.efi' failed
make: *** [/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp/DEBUG/SampleApp.efi]
 Error 134


build.py...
 : error 7000: Failed to execute command
        make tbuild [/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleApp]


build.py...
 : error F002: Failed to build module
        /home/u-uefi/src/edk2/SampleApp/SampleApp.inf [X64, GCC5, DEBUG]
```

The SampleApp.inf section `[LibraryClasses]` did not reference `UefiApplicationEntryPoint`

tianocore

Possible Build Errors

Error at the Shell prompt



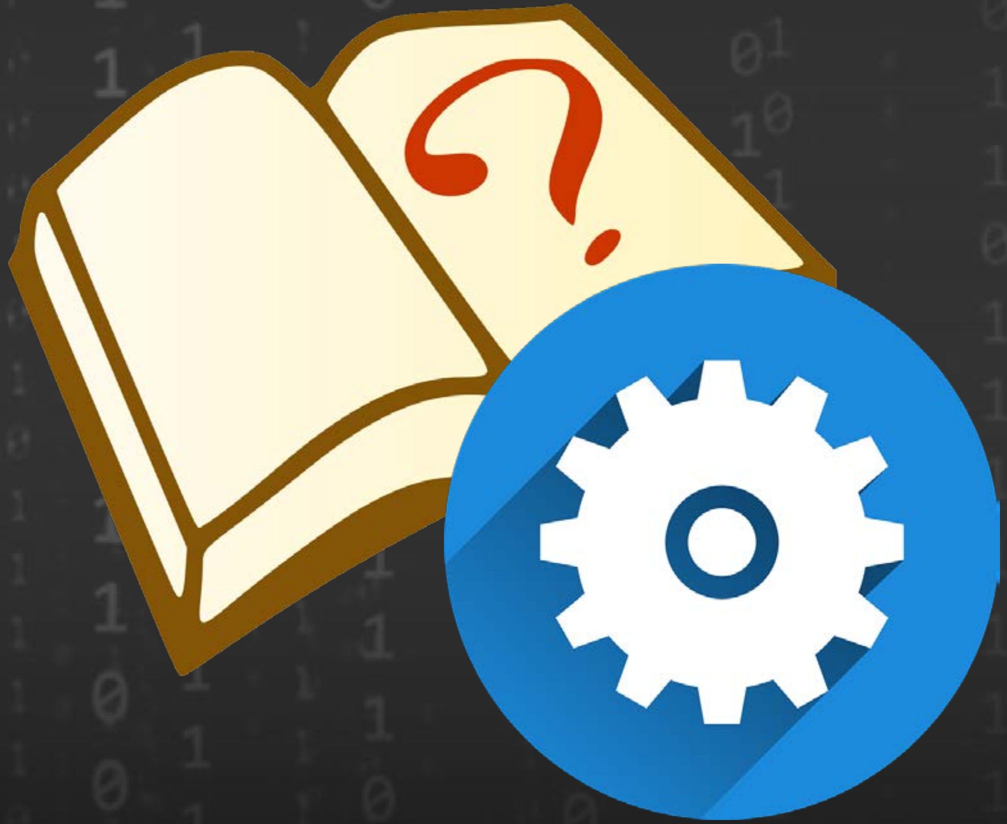
```
Press ESC in 4 seconds to skip startup.nsh or any other key to continue.
2.0 Shell> SampleApp
'SampleApp' is not recognized as an internal or external command, operable progr
am, or script file.
2.0 Shell> FS0:
2.0 FS0:\> LS SampleApp.efi
Error. No matching files were found.
2.0 FS0:\> _
```

Ensure the SampleApp.inf BaseName is SampleApp

tianocore

Lab 2.1: Build Switches

In this lab, you'll change the build switch ADD_SHELL_STRING to be always TRUE

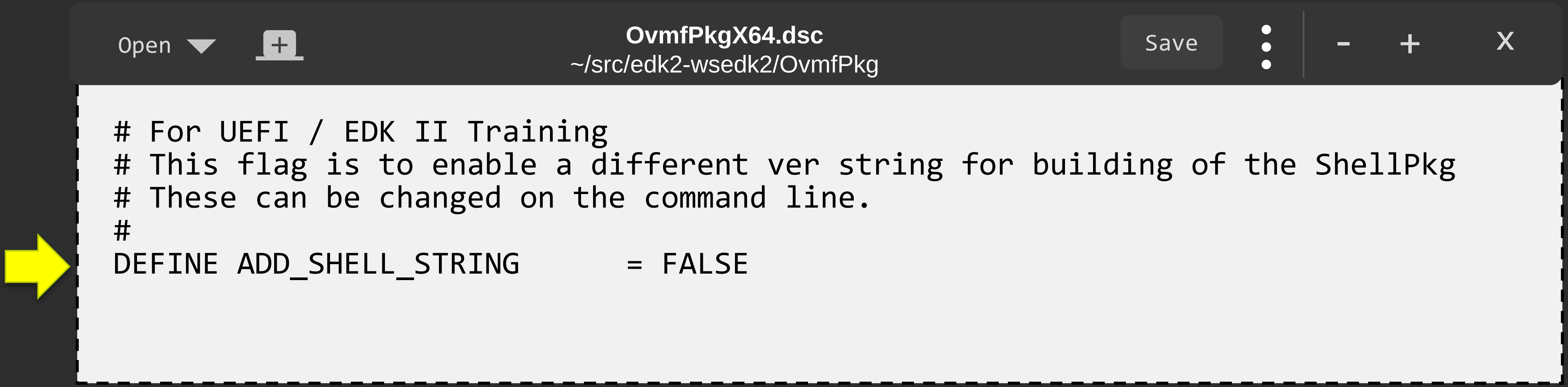


Build MACRO Switches

The build for `OvmfPkg` is using build MACRO Switch:

`-D ADD_SHELL_STRING` – used to change a string in the UEFI Shell application, only used for EDK II Training (requires ShellPkg be re-built on a change of this switch)

Edit ~/src/edk2-ws/edk2/OvmfPkg/OvmfPkgX64.dsc



```
# For UEFI / EDK II Training
# This flag is to enable a different ver string for building of the ShellPkg
# These can be changed on the command line.
#
DEFINE ADD_SHELL_STRING      = FALSE
```

tianocore

Lab 2.1: Compiling w/out Build Switch

Build SampleApp without the –D Switch

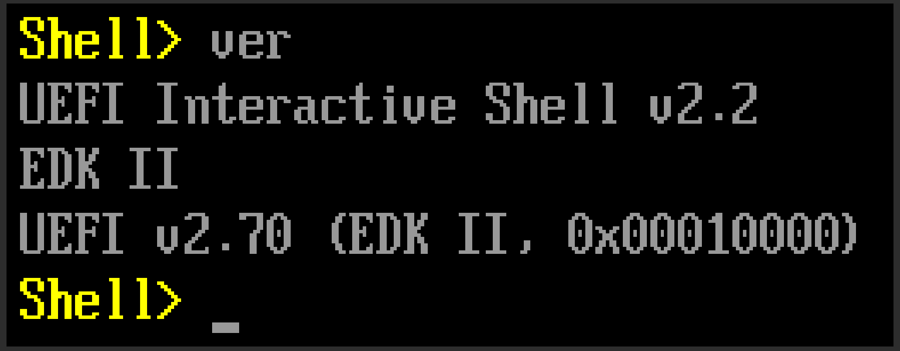
```
bash$ build
```

Copy OVMF.fd to run-ovmf

```
 bash$ cd ~/run-ovmf/
 bash$ cp ~/src/Build/OvmfX64/DEBUG_GCC5/FV/OVMF.fd bios.bin
```

Test by Invoking Qemu

```
 bash$ cd ~/run-ovmf
 bash$ . RunQemu.sh
```



```
Shell> ver
UEFI Interactive Shell v2.2
EDK II
UEFI v2.70 (EDK II, 0x00010000)
Shell> _
```

Check the Shell Version with the “Ver” command

NOTE: First delete directory   Build/OvmfPkgX64/DEBUG_GCC5/X64/ShellPkg
Exit QEMU

Lab 2.1: Compiling with Build Switch

Build SampleApp with the –D Switch

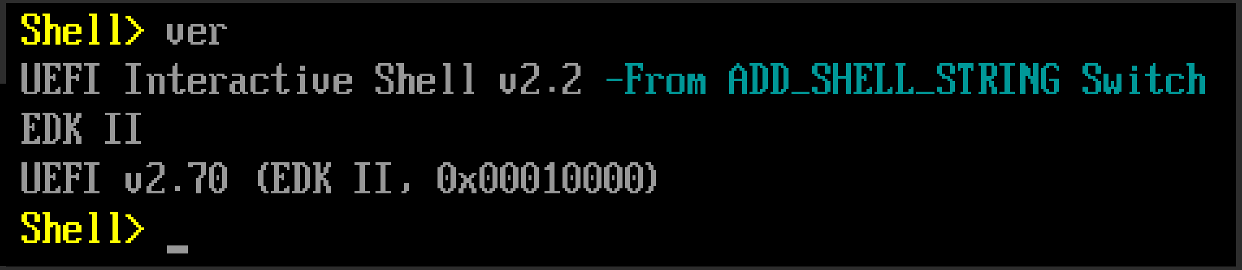
```
bash$ build -D ADD_SHELL_STRING
```

Copy OVMF.fd to run-ovmf

```
bash$ cd ~/run-ovmf/
bash$ cp ~/src/edk2-ws/Build/OvmfX64/DEBUG_GCC5/FV/OVMF.fd bios.bin
```

Test by Invoking Qemu

```
bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```



```
Shell> ver
UEFI Interactive Shell v2.2 -From ADD_SHELL_STRING Switch
EDK II
UEFI v2.70 (EDK II, 0x00010000)
Shell> _
```

Check the Shell Version with the “Ver” command – see the differences

Exit QEMU

tianocore

Lab 2.1: Compiling w/out Build Switch

Edit the file ~/src/edk2-ws/edk2/OvmfPkg/OvmfPkgX64.dsc

Change the `DEFINE ADD_SHELL_STRING = FALSE` to "`TRUE`" (appx. Line 31)

Build again

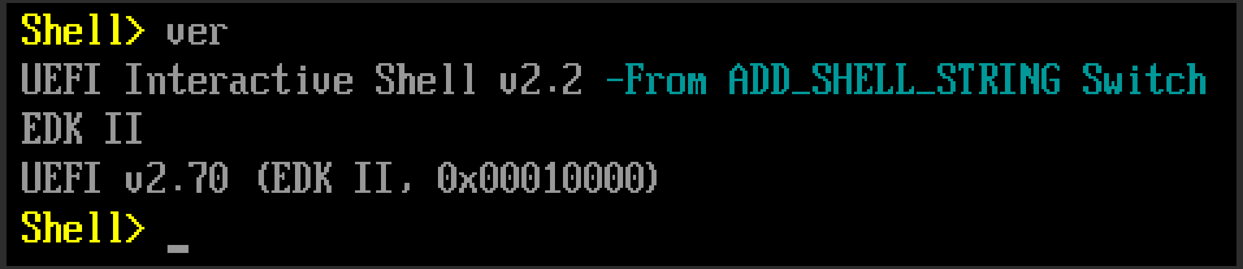
```
bash$ build
```

Copy OVMF.fd to run-ovmf

```
bash$ cd ~/run-ovmf/
bash$ cp ~/src/edk2-ws/Build/OvmfX64/DEBUG_GCC5/FV/OVMF.fd bios.bin
```

Test by Invoking Qemu

```
bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```



```
Shell> ver
UEFI Interactive Shell v2.2 -From ADD_SHELL_STRING Switch
EDK II
UEFI v2.70 (EDK II, 0x00010000)
Shell> _
```

Check the Shell version with "Ver" command

Exit QEMU

tianocore

Knowledge Check from LAB 2

1. How to write a simple native UEFI Application
2. Each module requires a .inf file with a unique GUID (use http://www.guidgenerator.com/)
3. The module created will be the base name defined in the .inf file
4. The module's .inf file is required to be included in the platform .dsc file
5. The [Packages] section is required at minimum to include MdePkg/MdePkg.dec
6. When using a Build Switch (-D) on the command line it overrides the value in the .DSC file
7. If it is a Library is getting updated, it is required to Build clean or delete the previous built module(s) including the library depending on what is getting re-built.

tianocore

Lab 2: If there are build errors …

See class files for the solution

- `...FW/LabSampleCode/LessonB.2`
- Copy the .inf and .c files to `~src/edk2-ws/edk2/SampleApp`
- Search sample DSC for reference to `SampleApp.inf` and add this line to your workspace DSC file `~src/edk2-ws/edk2/OvmfPkg/OvmfPkgX64.dsc`

Invoke `build` again and check the solution

ADD FUNCTIONALITY

Add Functionality to the Simple UEFI Application :
Next 3 Labs

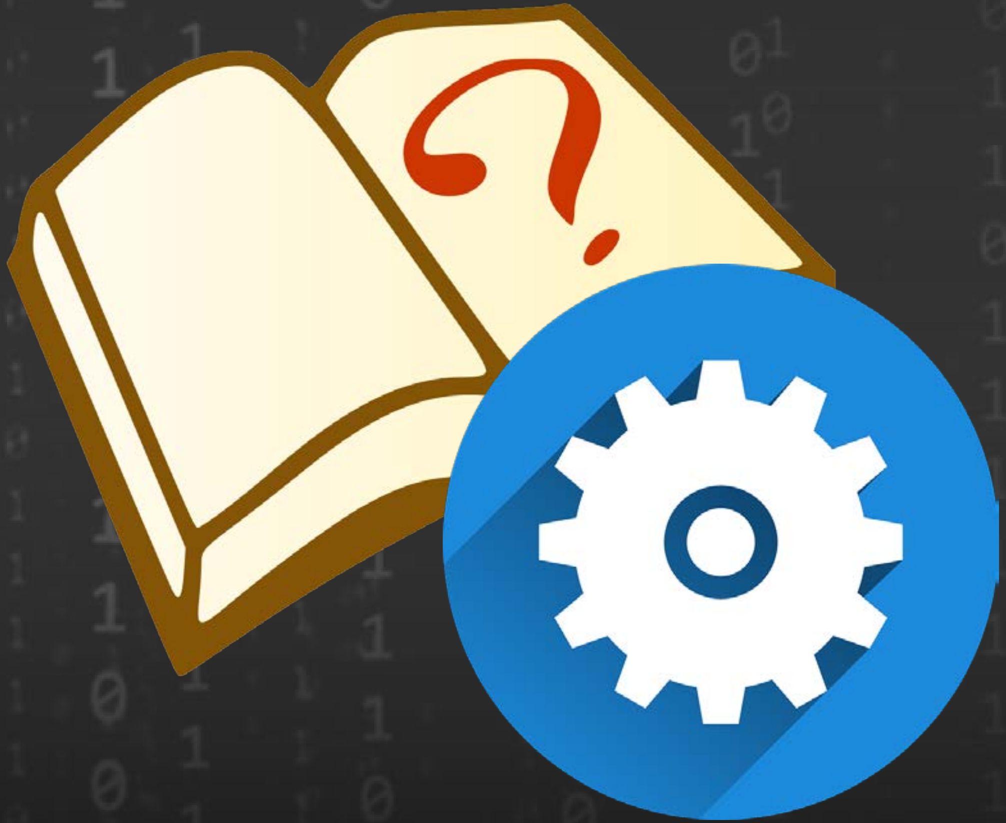
Lab 3:	Print the UEFI System Table
Lab 4:	Wait for an Event
Lab 5:	Create a Simple Typewriter function

Solutions in .../FW/LabSampleCode/LabSolutions/LessonB.n

tianocore

Lab 3: Print the UEFI System Table

Add code to print the hex address of the EFI System Table pointer to the console.

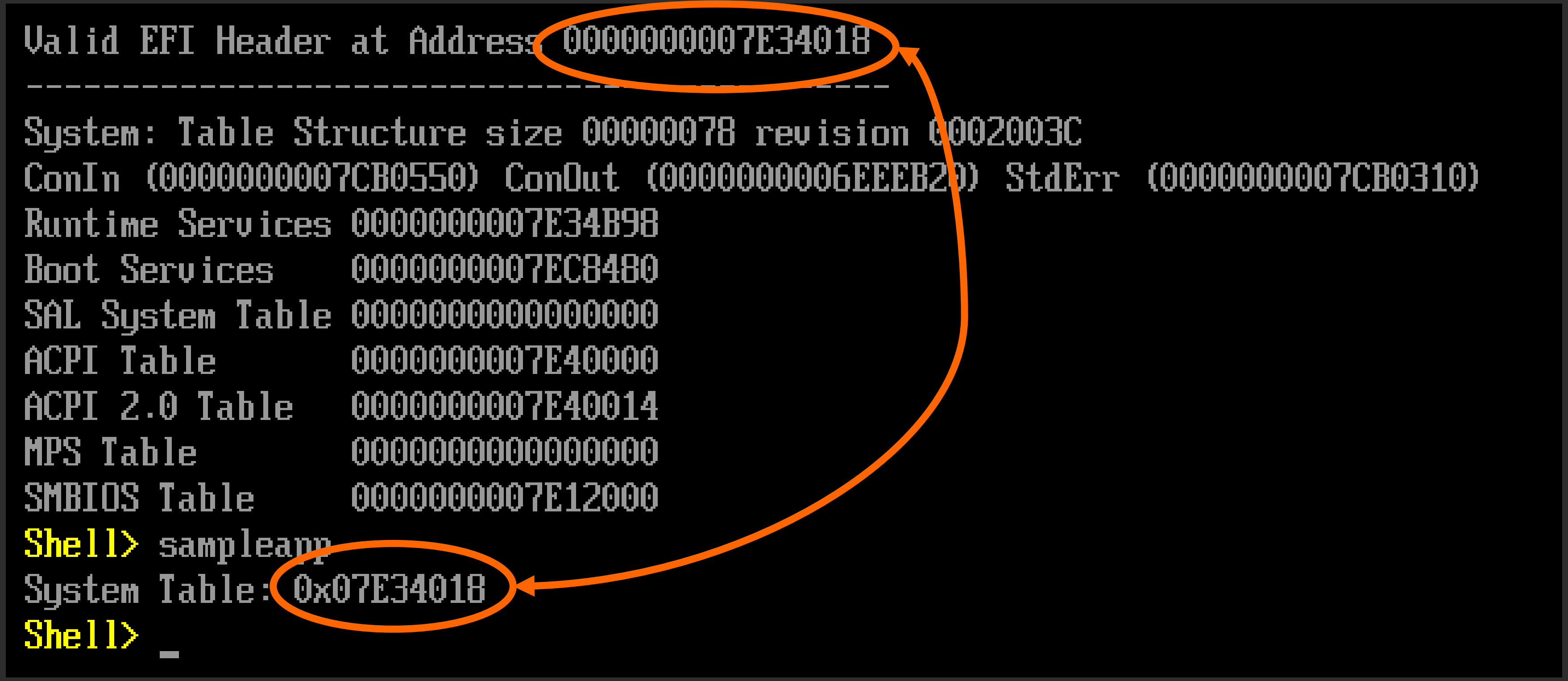


tianocore

Lab 3 : Add System Table Code

Add code to print to the console the hex address of the system table pointer

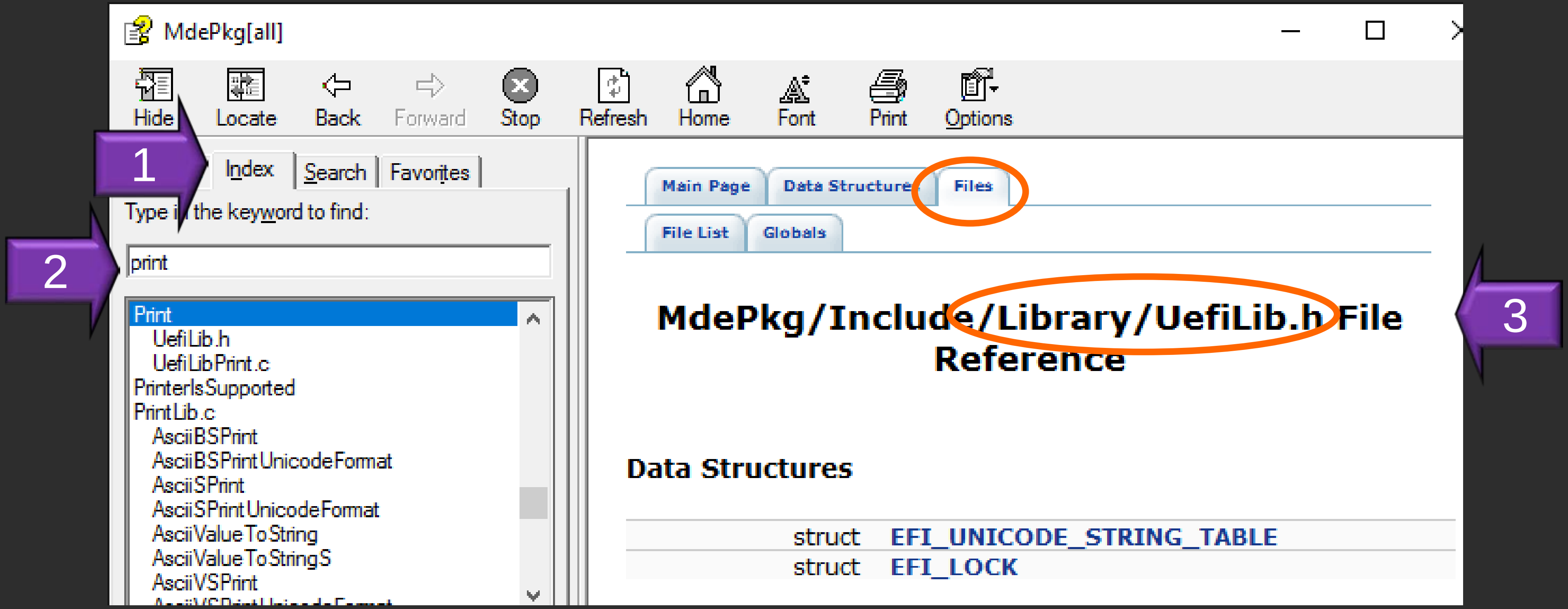
- Where is the “print” function?
- Where does the app get the pointer value?
  (compared to **`mem`** command below)



```
Valid EFI Header at Address 0000000007E34018
---------------------------------------------
System: Table Structure size 00000078 revision 0002003C
ConIn (0000000007CB0550) ConOut (0000000006EEEB20) StdErr (0000000007CB0310)
Runtime Services 0000000007E34B98
Boot Services    0000000007EC8480
SAL System Table 0000000000000000
ACPI Table       0000000007E40000
ACPI 2.0 Table   0000000007E40014
MPS Table        0000000000000000
SMBIOS Table     0000000007E12000
Shell> sampleapp
System Table: 0x07E34018
Shell> _
```

Lab 3 : Locating the Print() Function

1. Search the MdePkg.chm and find that the Print function by clicking on the "Index" tab
2. Type "Print" and double click
3. Scroll to the top in the right window to see that the print function is in the UefiLib.h file



Print
UefiLib.h
UefiLibPrint.c
PrinterIsSupported
PrintLib.c
AsciiBSPrint
AsciiBSPrintUnicodeFormat
AsciiSPrint
AsciiSPrintUnicodeFormat
AsciiValueToString
AsciiValueToStringS
AsciiVSPrint

MdePkg/Include/Library/UefiLib.h File Reference

Data Structures

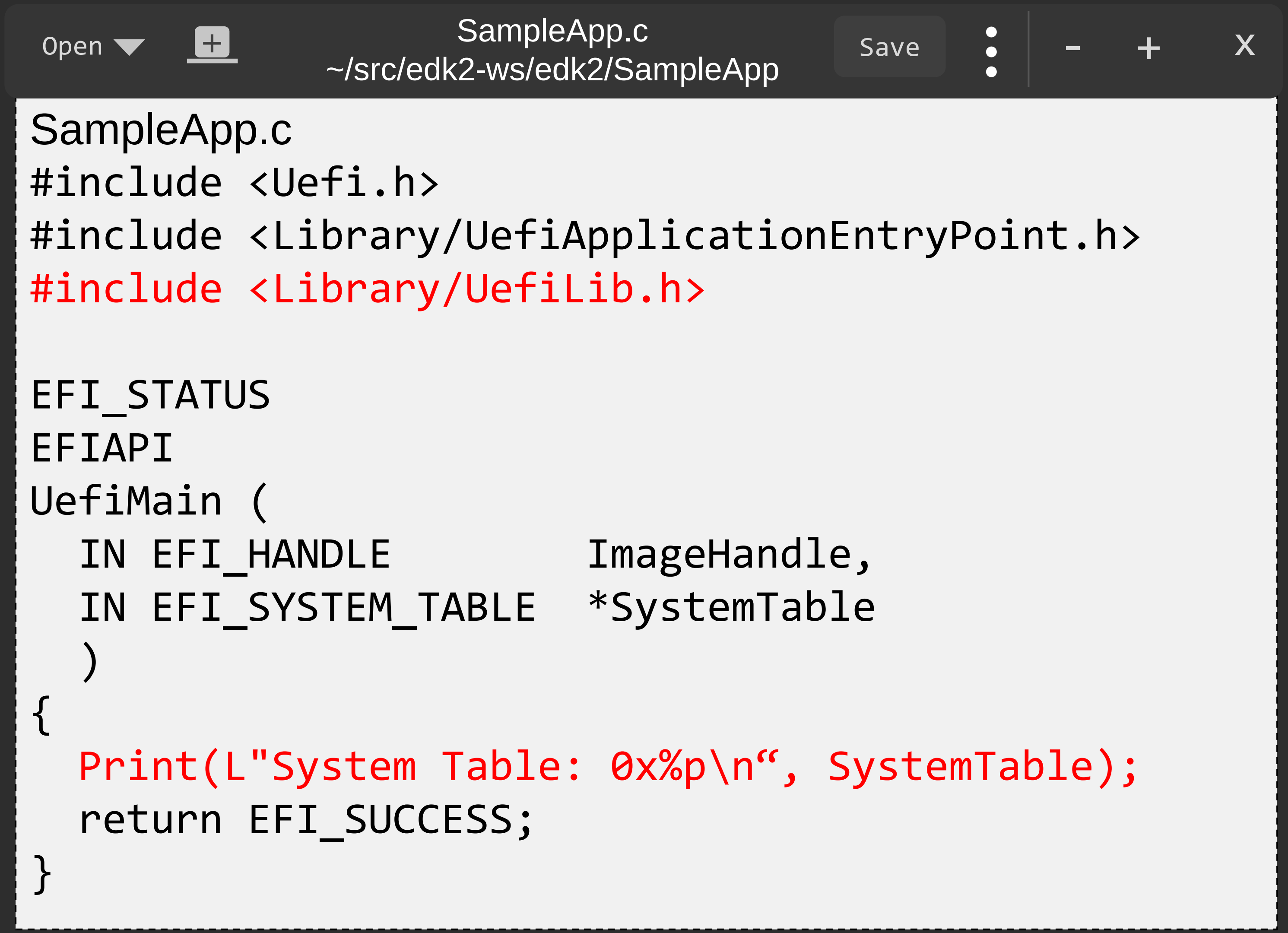
struct EFI_UNICODE_STRING_TABLE
struct EFI_LOCK

* NOTE: Install a CHM Viewer for Ubuntu - Clear Linux* Project See link for .chm viewer

```
bash$ sudo aptitude install kchmviewer
```

tianocore

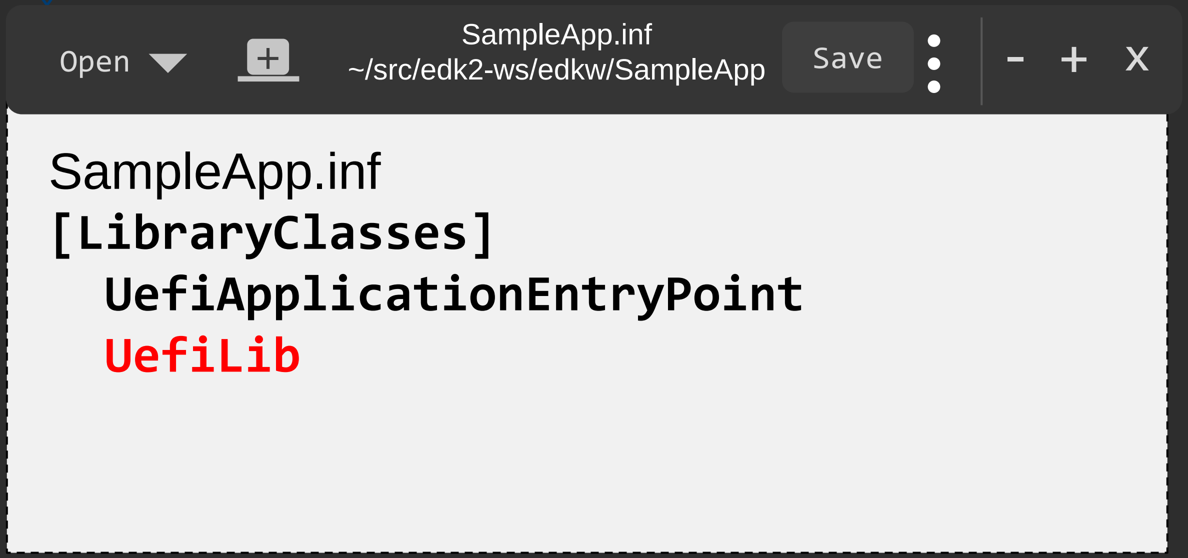
Lab 3 : Modifying .C & .INF Files



SampleApp.c
~/src/edk2-ws/edk2/SampleApp

SampleApp.c

```
#include <Uefi.h>
#include <Library/UefiApplicationEntryPoint.h>
#include <Library/UefiLib.h>

EFI_STATUS
EFIAPI
UefiMain (
  IN EFI_HANDLE        ImageHandle,
  IN EFI_SYSTEM_TABLE  *SystemTable
  )
{
  Print(L"System Table: 0x%p\n“, SystemTable);
  return EFI_SUCCESS;
}
```



SampleApp.inf
~/src/edk2-ws/edkw/SampleApp

SampleApp.inf

```
[LibraryClasses]
  UefiApplicationEntryPoint
  UefiLib
```

Note: Solution files are in the lab materials directory

tianocore

Lab 3 : Build and Test SampleApp

Build SampleApp – Cd to ~/src/edk2-ws/edk2 directory

```
bash$ build
```

Copy SampleApp.efi to hda-contents

```
bash$ cd ~/run-ovmf/hda-contents
bash$ cp ~/src/edk2-ws/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp.efi .
```

Test by Invoking Qemu

```
bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```

Run the application from the shell

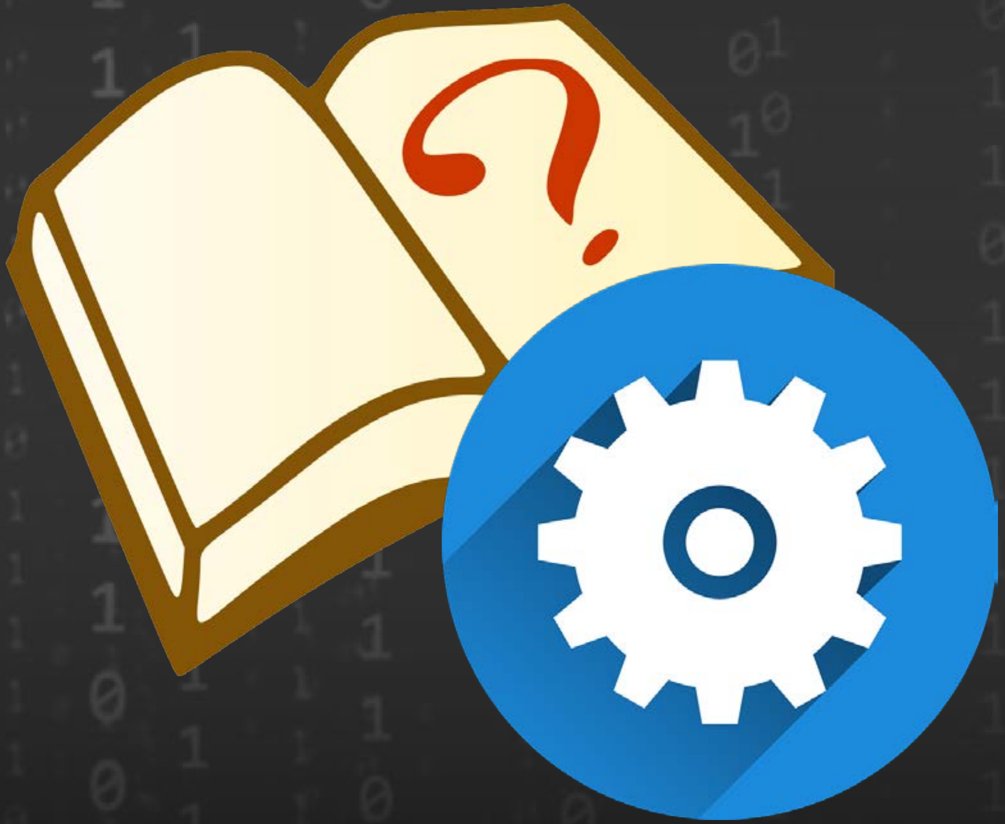
```
Shell> SampleApp
System Table: 0x07E34018
```

Exit QEMU

tianocore

Lab 4: Waiting for an Event

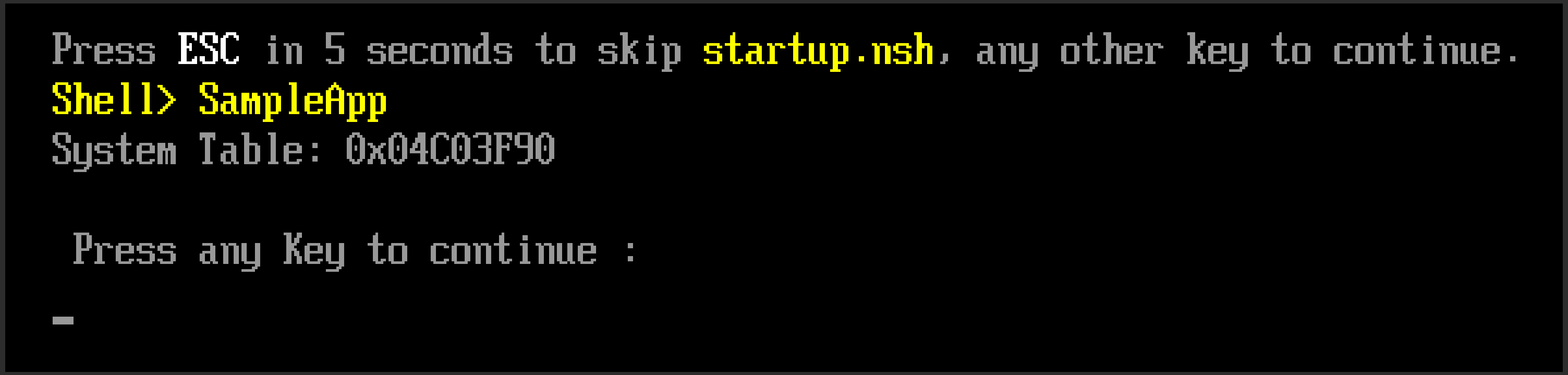
In this lab, you'll learn how to locate code and .chm files to help write EFI code for waiting for an event



tianocore

Lab 4 : Add Wait for Event

Add code to make your application wait for a key press event (`WaitForEvent` / `WaitForKey`)



```
Press ESC in 5 seconds to skip startup.nsh, any other key to continue.
Shell> SampleApp
System Table: 0x04C03F90

 Press any Key to continue :

_
```

- Where are these functions located?
- What else can you do with the key press?

tianocore

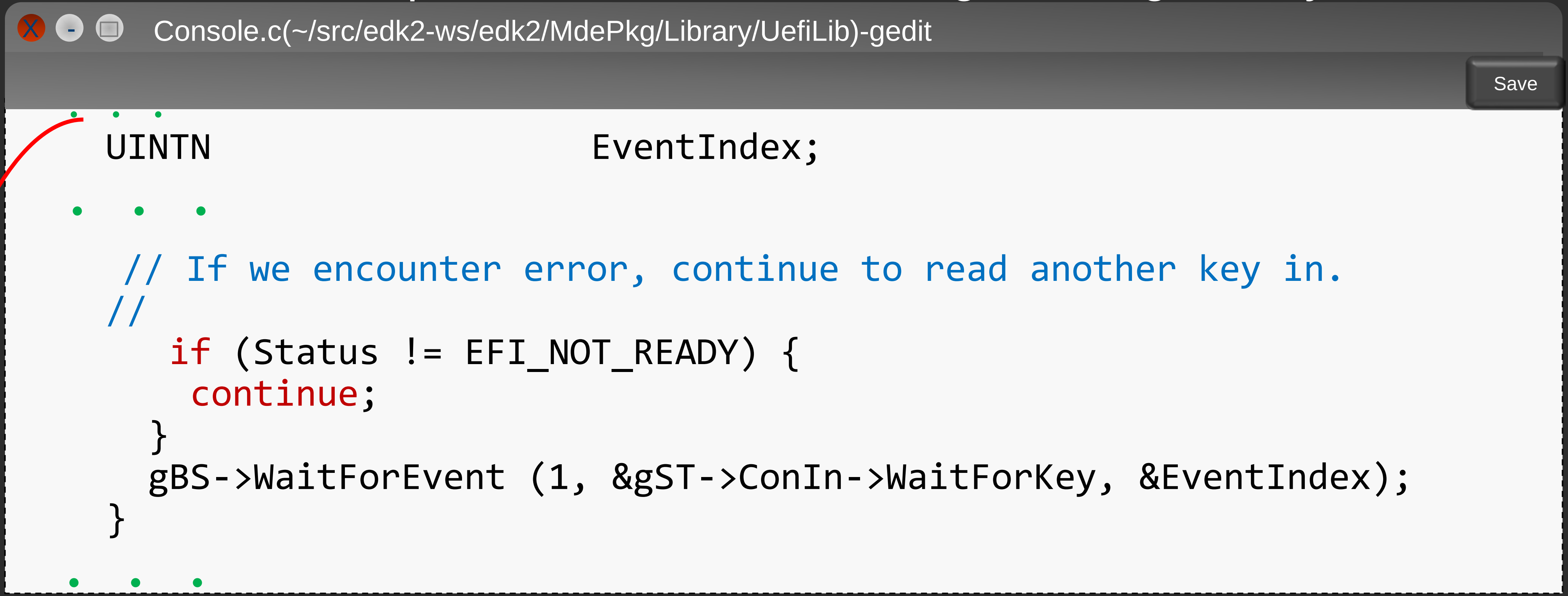
Lab 4: HOW?

Locate Functions: `WaitForEvent / WaitForKey`

- Search MdePkg.chm- "MdePkg Document With Libraries.chm" located in ... Lab_Material_FW/FW/Documentation
  - Locate `WaitForEvent` in Boot Services
  - Locate `WaitForKey` and find (`EFI_SIMPLE_TEXT_INPUT_PROTOCOL` will be part of ConIn)
- Check the UEFI Spec for parameters needed:
  - `WaitForEvent` is referenced via Boot Services pointer, which is referenced via EFI System Table
  - `WaitForKey` can be referenced through the EFI System Table passed into the application
- **OR** Search the working space for `WaitForEvent` for an example
- One can be found in MdePkg/Library/UefiLib/Console.c ~ ln 569:

Lab 4 : Update the C File for `WaitForKey`

Search the work space and find the following MdePkg/Library/UefiLib/Console.c ~ ln 563:



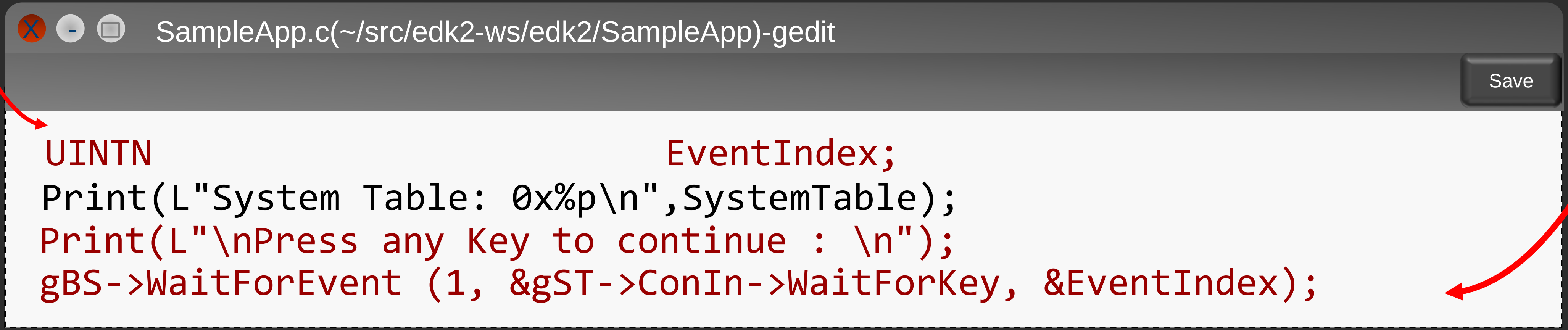
Console.c(~/src/edk2-ws/edk2/MdePkg/Library/UefiLib)-gedit

```
. . .
UINTN                       EventIndex;
 . . .
  // If we encounter error, continue to read another key in.
  //
     if (Status != EFI_NOT_READY) {
      continue;
    }
    gBS->WaitForEvent (1, &gST->ConIn->WaitForKey, &EventIndex);
  }
 . . .
```

Line 410

Line 563

Add the following to `SampleApp.c`



SampleApp.c(~/src/edk2-ws/edk2/SampleApp)-gedit

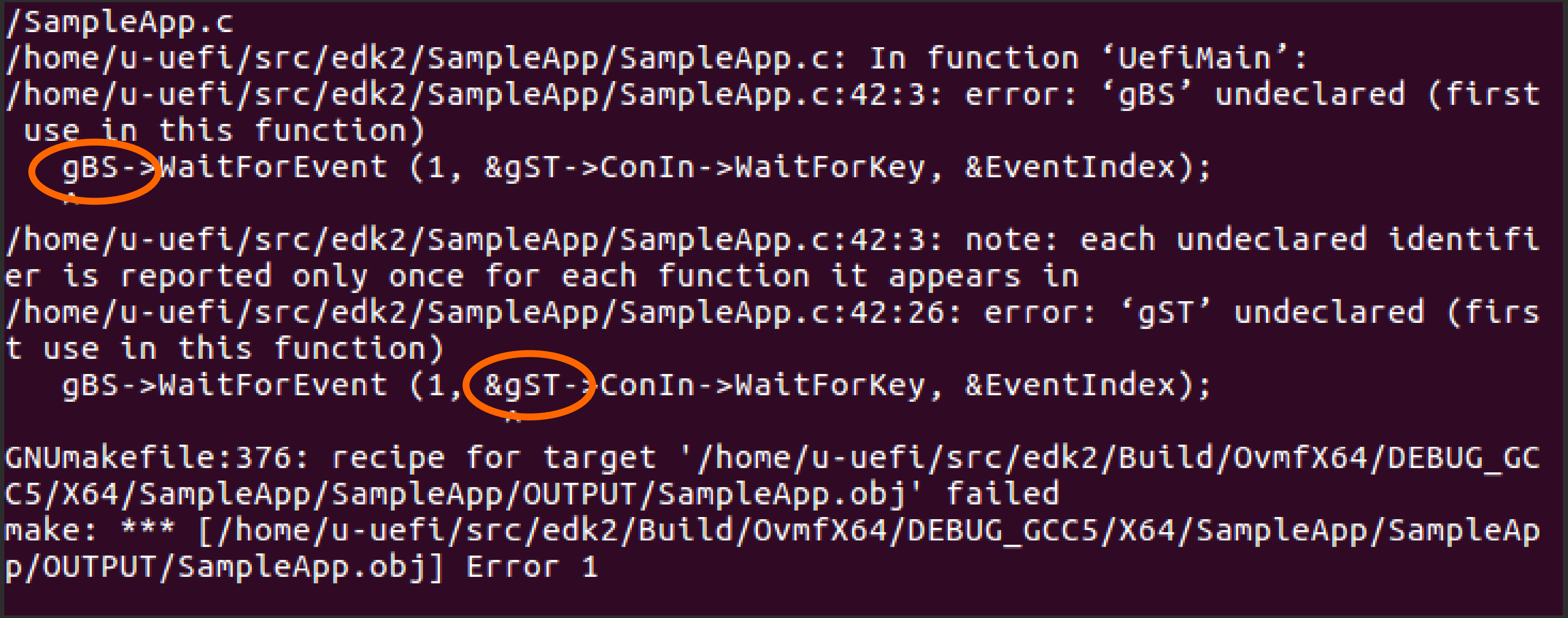
```
UINTN                            EventIndex;
Print(L"System Table: 0x%p\n",SystemTable);
Print(L"\nPress any Key to continue : \n");
gBS->WaitForEvent (1, &gST->ConIn->WaitForKey, &EventIndex);
```

Copy and Paste

tianocore

Lab 4: Test Compile

However, this won't compile … `gBS` and `gST` are not defined.



```
/SampleApp.c
/home/u-uefi/src/edk2/SampleApp/SampleApp.c: In function ‘UefiMain’:
/home/u-uefi/src/edk2/SampleApp/SampleApp.c:42:3: error: ‘gBS’ undeclared (first
 use in this function)
   gBS->WaitForEvent (1, &gST->ConIn->WaitForKey, &EventIndex);
   ^
/home/u-uefi/src/edk2/SampleApp/SampleApp.c:42:3: note: each undeclared identifi
er is reported only once for each function it appears in
/home/u-uefi/src/edk2/SampleApp/SampleApp.c:42:26: error: ‘gST’ undeclared (firs
t use in this function)
   gBS->WaitForEvent (1, &gST->ConIn->WaitForKey, &EventIndex);
                          ^
GNUmakefile:376: recipe for target '/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GC
C5/X64/SampleApp/SampleApp/OUTPUT/SampleApp.obj' failed
make: *** [/home/u-uefi/src/edk2/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp/SampleAp
p/OUTPUT/SampleApp.obj] Error 1
```

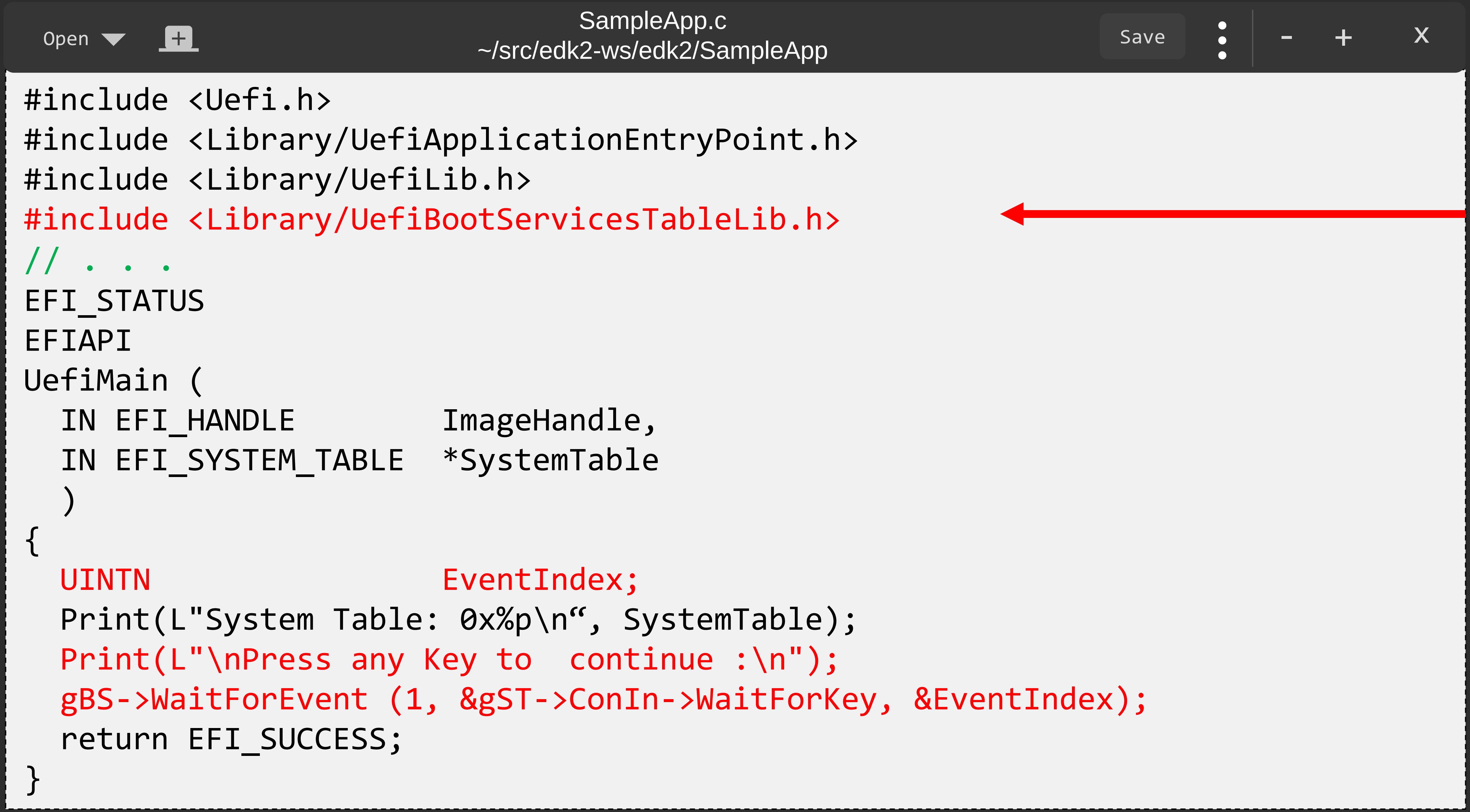
Search the MdePkg.chm for "`gBS`" and "`gST`" – they are located in `UefiBootServicesTableLib.h`

Add the boot services lib to **`SampleApp.c`** …

```
#include <Library/UefiBootServicesTableLib.h>
```

(hint: **`Lesson B.4`** has the solution)

Lab 4: Update for gBS & gST



```c
#include <Uefi.h>
#include <Library/UefiApplicationEntryPoint.h>
#include <Library/UefiLib.h>
#include <Library/UefiBootServicesTableLib.h>
// . . .
EFI_STATUS
EFIAPI
UefiMain (
  IN EFI_HANDLE        ImageHandle,
  IN EFI_SYSTEM_TABLE  *SystemTable
  )
{
  UINTN                EventIndex;
  Print(L"System Table: 0x%p\n“, SystemTable);
  Print(L"\nPress any Key to  continue :\n");
  gBS->WaitForEvent (1, &gST->ConIn->WaitForKey, &EventIndex);
  return EFI_SUCCESS;
}
```

tianocore

Lab 4 : Build and Test SampleApp

Build SampleApp – Cd to ~/src/edk2-ws/edk2 directory

```
bash$ build
```

Copy SampleApp.efi to hda-contents

```
 bash$ cd ~/run-ovmf/hda-contents
 bash$ cp ~/src/edk2-ws/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp.efi .
```

Test by Invoking Qemu

```
 bash$ cd ~/run-ovmf
 bash$ . RunQemu.sh
```

Run the application from the shell

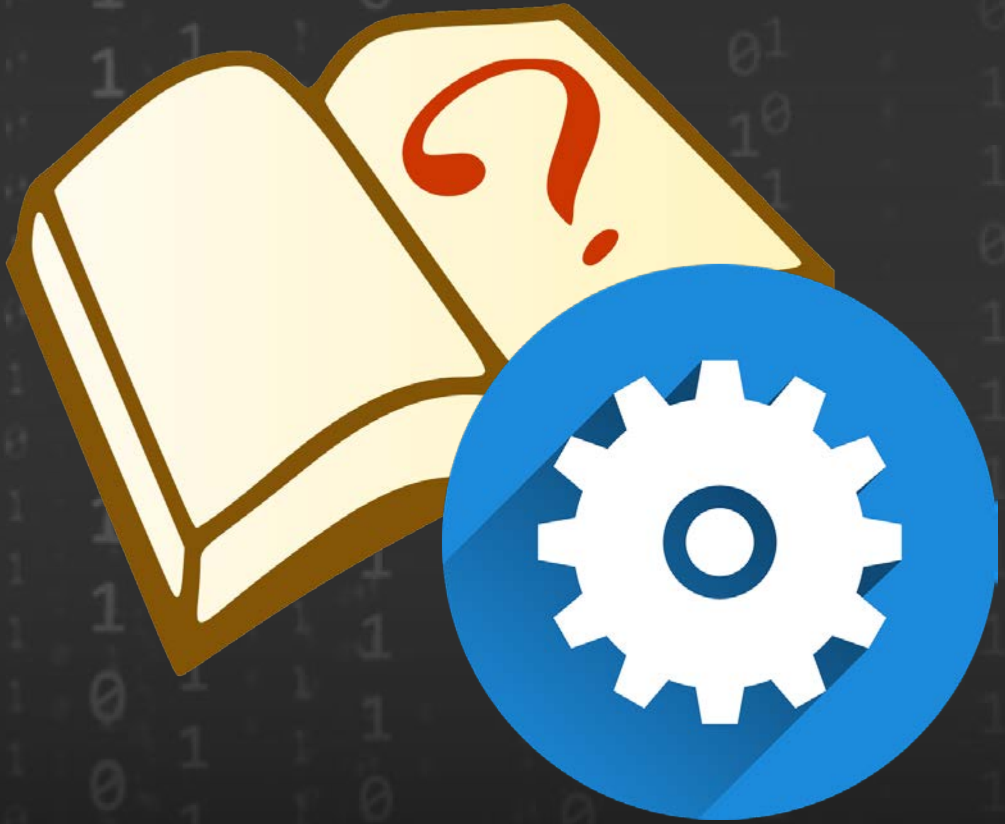
```
     Shell> SampleApp
     System Table: 0x07E34018

     Press any key to continue:
```

Notice that the SampleApp will wait until a key press to continue.
Exit QEMU

tianocore

Lab 5: Creating a Simple Typewriter Function

In this lab, you'll learn how to create a simple typewriter function that retrieves the keys you type and subsequently prints each one back to the console

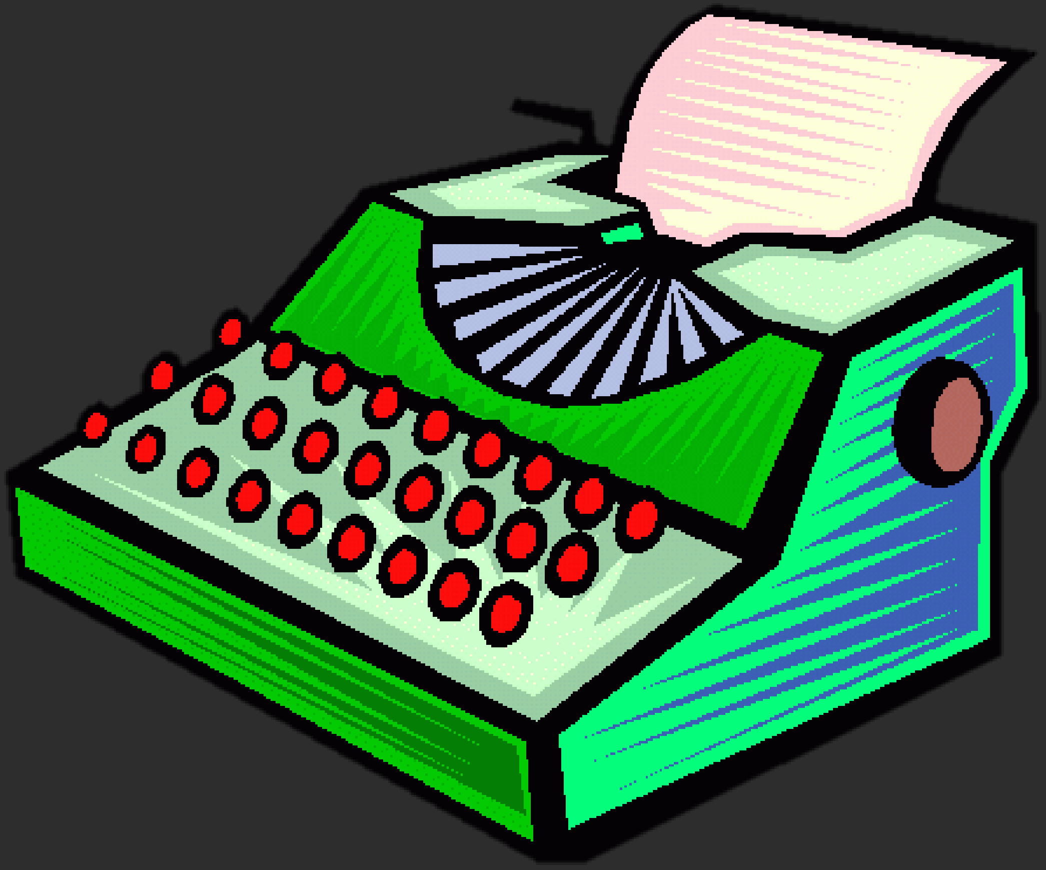


Lab 5 : Typewriter Function

Create a Simple Typewriter Function using the SampleApp from Lab 4

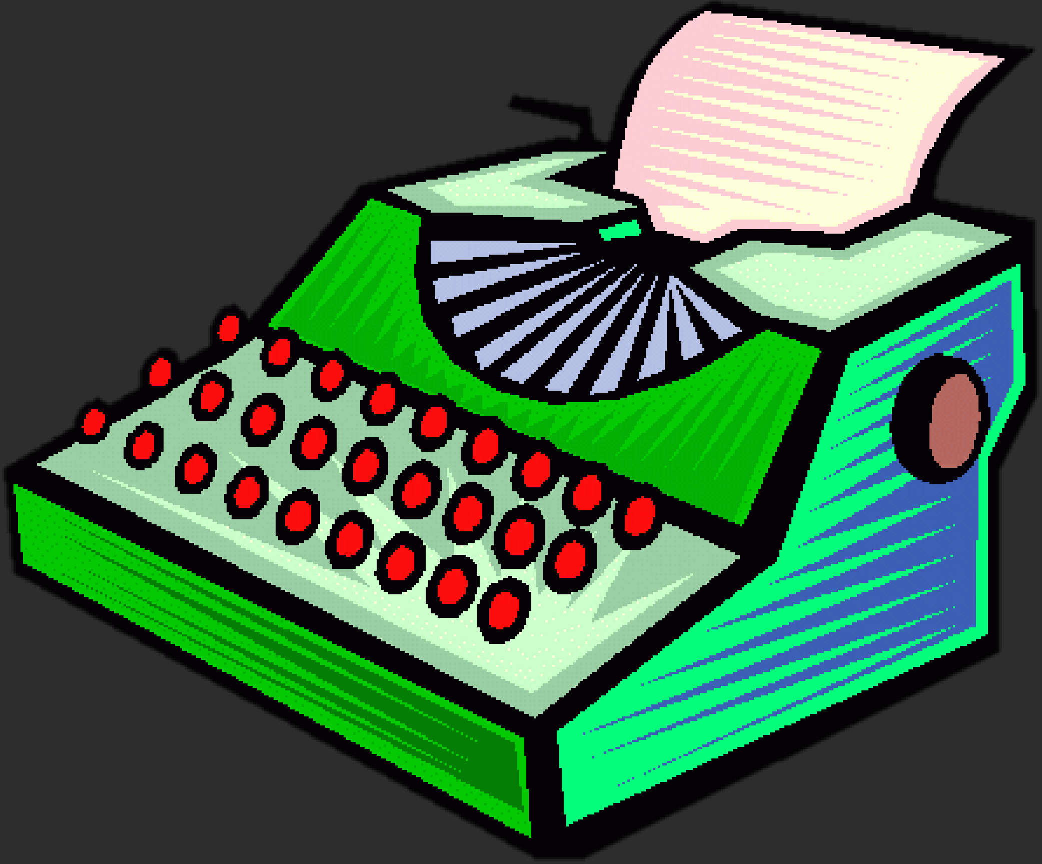
Requirements:

- Retrieve keys entered from keyboard (*Like* Lab 4)
- Print back each key entered to the console
- To exit, press "." (DOT) and then <Enter>



Lab 5 : Typewriter Function

Create a Simple Typewriter Function using the SampleApp from Lab 4



How:

1. Add a Loop using `WaitForEvent` with `WaitForKey`
2. Use the `ReadKeyStroke` function from `ConIn`
3. Print back each key to console
4. Exit when DOT "." character is followed by an <Enter> key

tianocore

Lab 5: How Process (Hints)

- Use the same procedure as with Lab 4 to find "`ReadKeyStroke`" in the workspace: MdePkg/Library/UefiLib/Console.c ~ ln 552

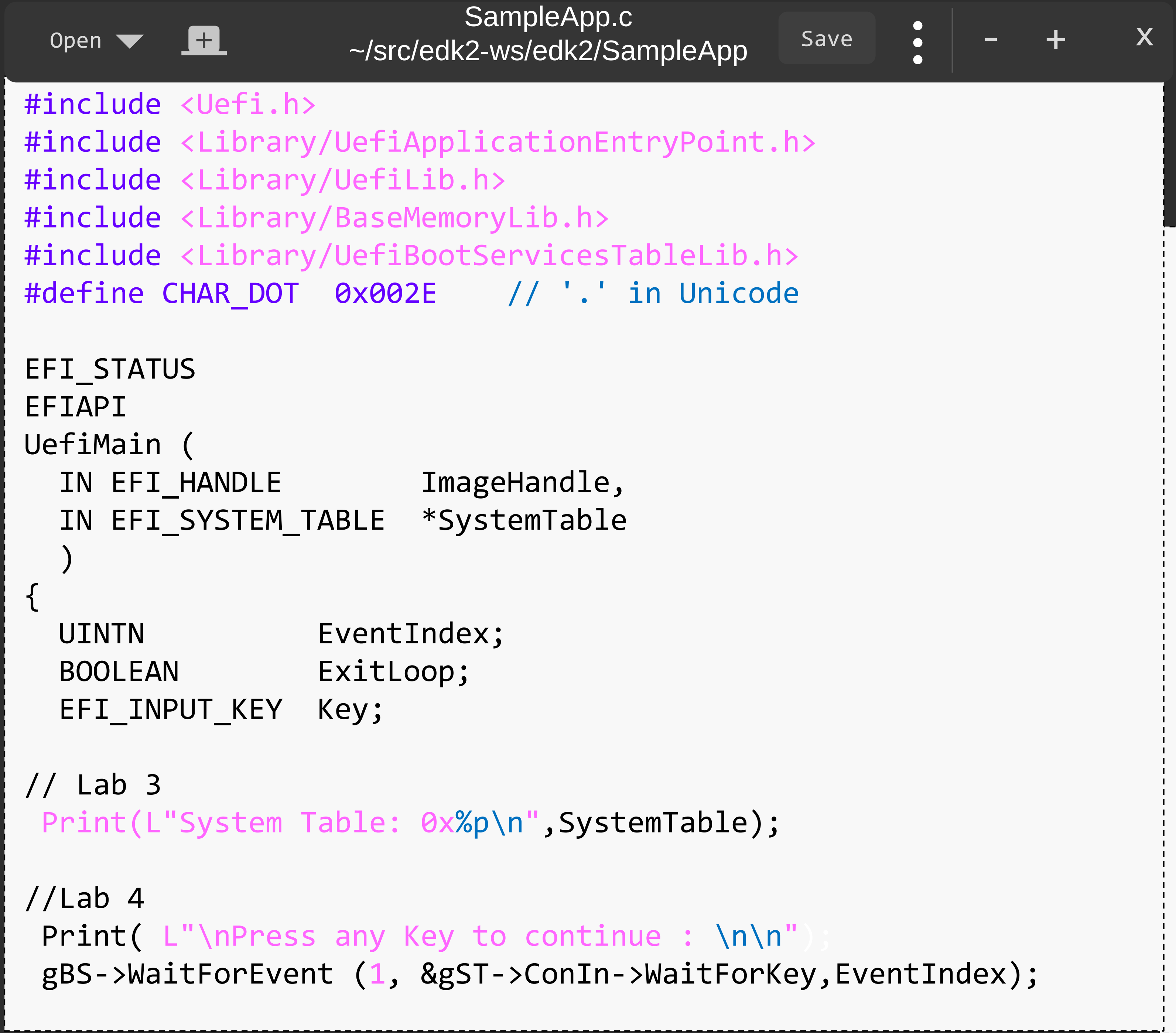
```
Status = gST->ConIn->ReadKeyStroke (gST->ConIn, Key);
```

- `ReadKeyStroke` uses buffer called `EFI_INPUT_KEY` ~ ln 393

```
OUT EFI_INPUT_KEY  *Key,
```

- TIP: Good Idea to zero out a buffer in your function –
  - Use MdePkg.chm to find `ZeroMem` function
  - Use `ZeroMem` on your variable buffer "`Key`" of type `EFI_INPUT_KEY`
- Use Boolean flag "`ExitLoop`" to exit your loop once the user enters a DOT "." character.

Lab 5: Solution



```c
#include <Uefi.h>
#include <Library/UefiApplicationEntryPoint.h>
#include <Library/UefiLib.h>
#include <Library/BaseMemoryLib.h>
#include <Library/UefiBootServicesTableLib.h>
#define CHAR_DOT  0x002E    // '.' in Unicode

EFI_STATUS
EFIAPI
UefiMain (
  IN EFI_HANDLE        ImageHandle,
  IN EFI_SYSTEM_TABLE  *SystemTable
  )
{
  UINTN          EventIndex;
  BOOLEAN        ExitLoop;
  EFI_INPUT_KEY  Key;

// Lab 3
 Print(L"System Table: 0x%p\n",SystemTable);

//Lab 4
 Print( L"\nPress any Key to continue : \n\n");
 gBS->WaitForEvent (1, &gST->ConIn->WaitForKey,EventIndex);
```

(hint: Lesson B.5 has the solution)

```c
// Lab 5
 Print(L"Enter text. Include a dot ('.') in a \
     sentence then <Enter> to exit:\n\n”);
 ZeroMem (&Key, sizeof (EFI_INPUT_KEY));
 gST->ConIn->ReadKeyStroke (gST->ConIn, &Key);
 ExitLoop = FALSE;
 do {
        gBS->WaitForEvent (1, &gST->ConIn->WaitForKey,
             &EventIndex);
        gST->ConIn->ReadKeyStroke (gST->ConIn, &Key);
        Print(L"%c", Key.UnicodeChar);
        if (Key.UnicodeChar == CHAR_DOT){
             ExitLoop = TRUE;
        }
    } while (!(Key.UnicodeChar == CHAR_LINEFEED  ||
       Key.UnicodeChar == CHAR_CARRIAGE_RETURN) ||
       !(ExitLoop) );

 Print(L"\n");
 return EFI_SUCCESS;
}
```

Lab 5: Solution

SampleApp.c Should have the following for Lab 5:

```
#include <Uefi.h>
#include <Library/UefiApplicationEntryPoint.h>
#include <Library/UefiLib.h>
#include <Library/UefiBootServicesTableLib.h>
#include <Library/BaseMemoryLib.h>
#define CHAR_DOT  0x002E    // '.' in Unicode

EFI_STATUS
EFIAPI
UefiMain (
  IN EFI_HANDLE        ImageHandle,
  IN EFI_SYSTEM_TABLE  *SystemTable
  )
{
  UINTN          EventIndex;
  BOOLEAN        ExitLoop;
  EFI_INPUT_KEY  Key;

// Lab 3
 Print(L"System Table: 0x%p\n",SystemTable);

//Lab 4
 Print( L"\nPress any Key to continue : \n\n");
 gBS->WaitForEvent (1, &gST->ConIn->WaitForKey,
```

```
// Lab 5
 Print(L"Enter text. Include a dot ('.') in a sentence then
<Enter> to exit:\n\n");
 ZeroMem (&Key, sizeof (EFI_INPUT_KEY));
 gST->ConIn->ReadKeyStroke (gST->ConIn, &Key);
 ExitLoop = FALSE;
 do {
       gBS->WaitForEvent (1, &gST->ConIn-
>WaitForKey,&EventIndex);
       gST->ConIn->ReadKeyStroke (gST->ConIn, &Key);
       Print(L"%c", Key.UnicodeChar);
       if (Key.UnicodeChar == CHAR_DOT){
            ExitLoop = TRUE;
       }
    } while (!(Key.UnicodeChar == CHAR_LINEFEED  ||
       Key.UnicodeChar == CHAR_CARRIAGE_RETURN) ||
       !(ExitLoop) );

 Print(L"\n");
 return EFI_SUCCESS;
}
```

tianocore

Lab 5 :Build and Test SampleApp

Build SampleApp – Cd to ~/src/edk2-ws/edk2 dir

```
bash$ build
```

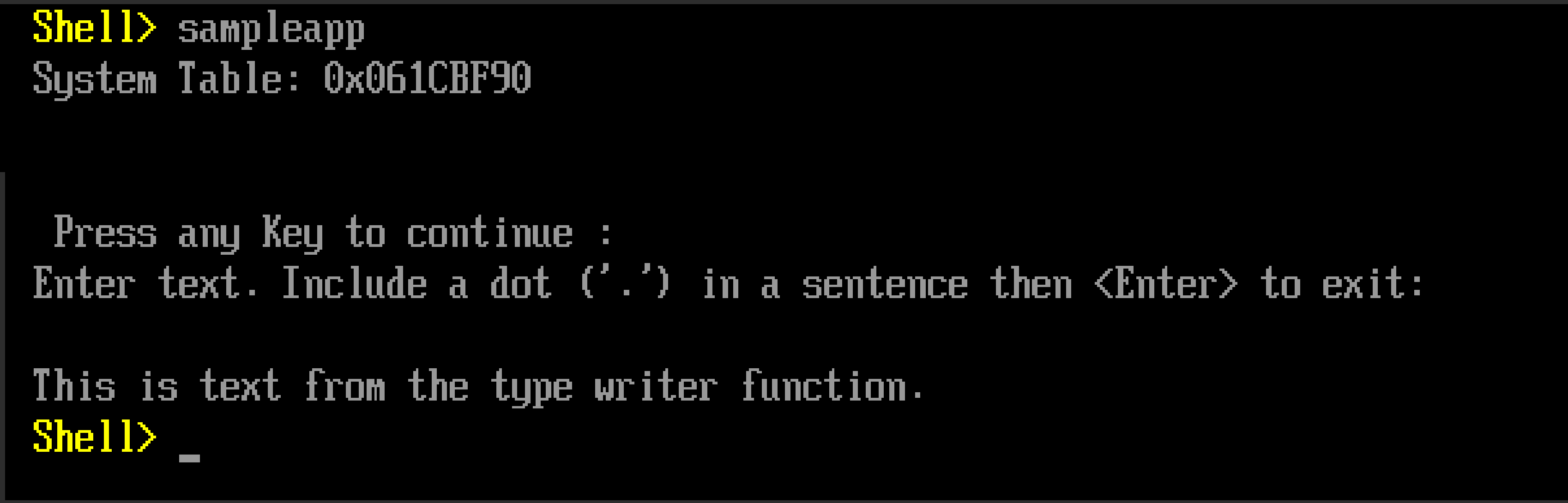
Copy SampleApp.efi to hda-contents

```
bash$ cd ~/run-ovmf/hda-contents
bash$ cp ~/src/edk2-ws/Build/OvmfX64/DEBUG_GCC5/X64/SampleApp.efi .
```

Test by Invoking Qemu

```
bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```

Run the application from the shell



```
Shell> sampleapp
System Table: 0x061CBF90


 Press any Key to continue :
Enter text. Include a dot ('.') in a sentence then <Enter> to exit:

This is text from the type writer function.
Shell> _
```

Exit QEMU

tianocore

Bonus Exercise: Open Protocol Example

Write an Application using **`argv, argc`** parameters

- Captures command line parameters using Open Protocol
- Need to open SHELL_INTERFACE_PROTOCOL
- Note: Requires ShellPkg

Build SampleApp – Cd to ~/src/edk2-ws/edk2

Copy SampleApp.efi to hda-contents

Test by Invoking Qemu

```
bash$ build


bash$ cp Build/OvmfX64/DEBUG_GCC5/X64/SampleApp.efi \
      ~/run-ovmf/hda-contents


bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```

Run the application from the shell

```
Shell> SampleApp  test1 test2
```

(hint: ~FW/LabSampleCode/ShellAppSample has the solution)

tianocore

USING EADK

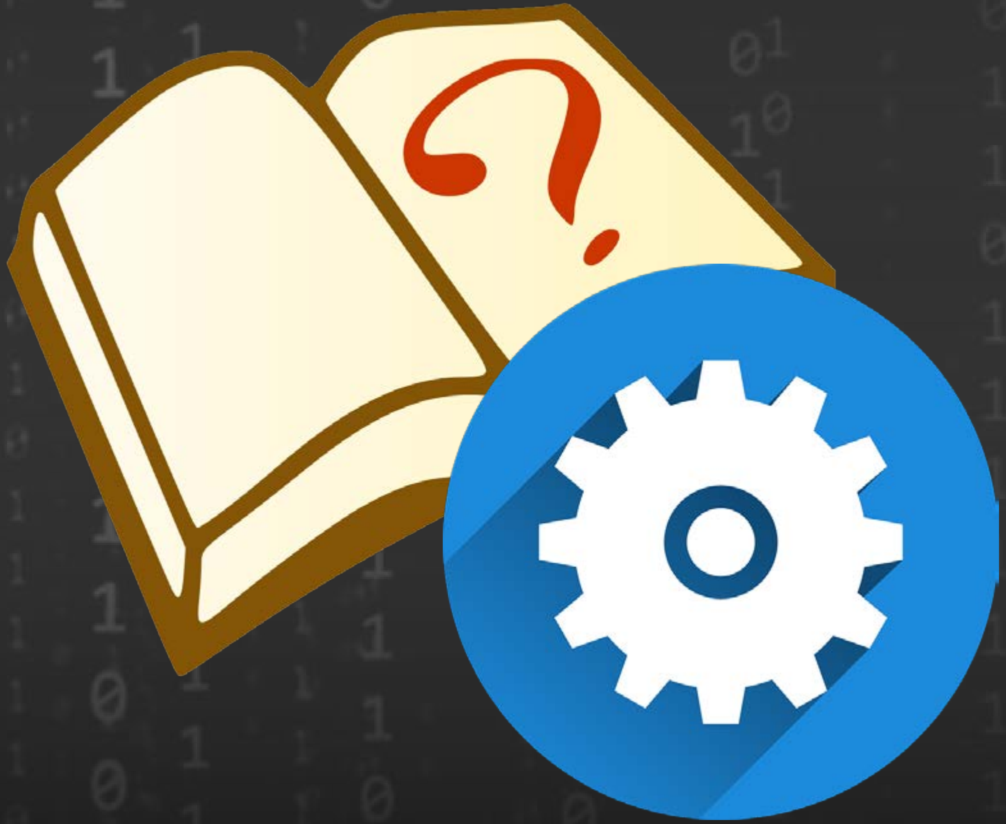
Using EADK with UEFI Application

Labs 6-7 are Optional

tianocore

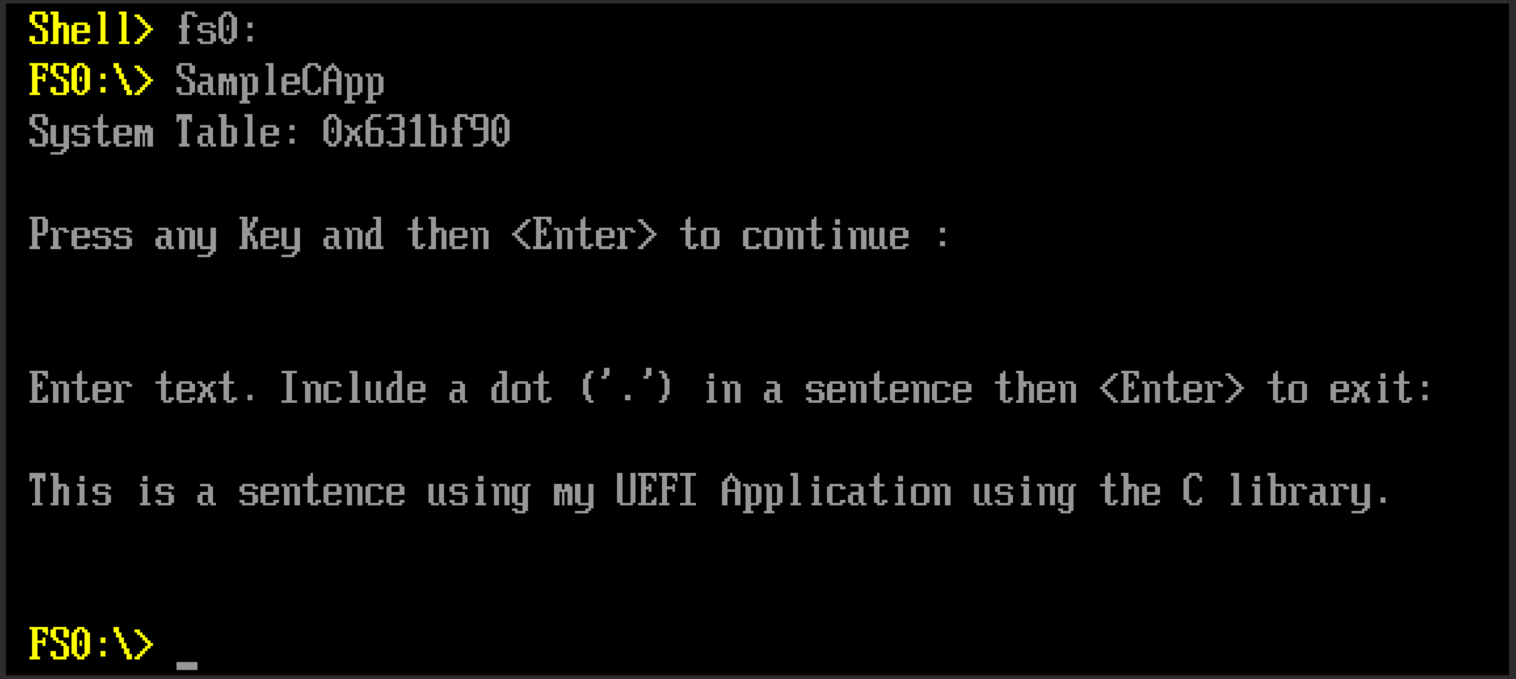
Lab 6: Writing UEFI Applications with EADK

In this lab, you'll write an application with the same functionality as SampleApp.c using LibC from the EDK II Application Development Kit (EADK)



Lab 6: With EDK II EADK

Write the same application with the same functionality as SampleApp.c using the LibC from the EADK



```
Shell> fs0:
FS0:\> SampleCApp
System Table: 0x631bf90

Press any Key and then <Enter> to continue :


Enter text. Include a dot ('.') in a sentence then <Enter> to exit:

This is a sentence using my UEFI Application using the C library.


FS0:\> _
```

What libraries are needed

What differences are there using the LibC

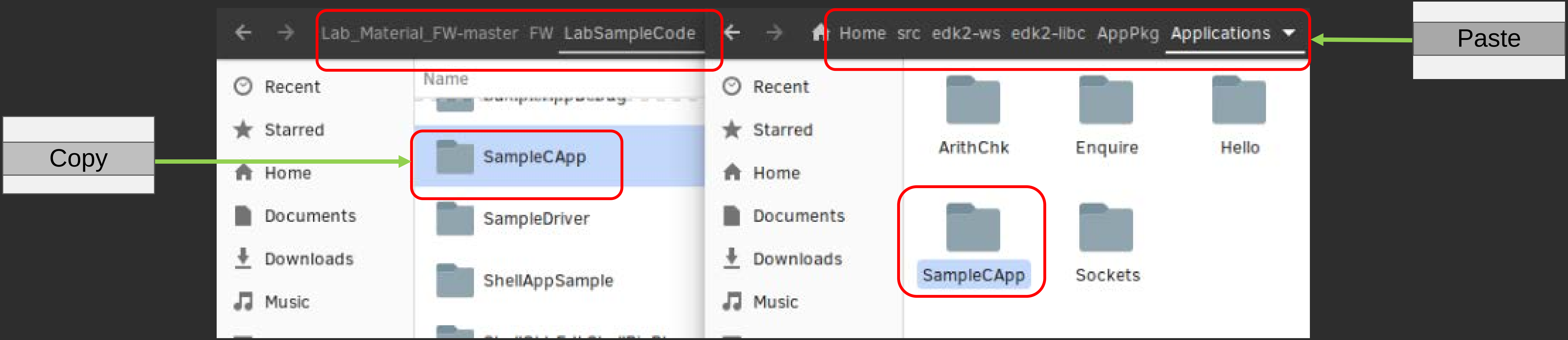
tianocore

Lab 6: EDK II using EADK

Start with the packages for EADK

- /edk2 - AppPkg - has directory Applications
- /edk2 - StdLib - contains the LibC libraries

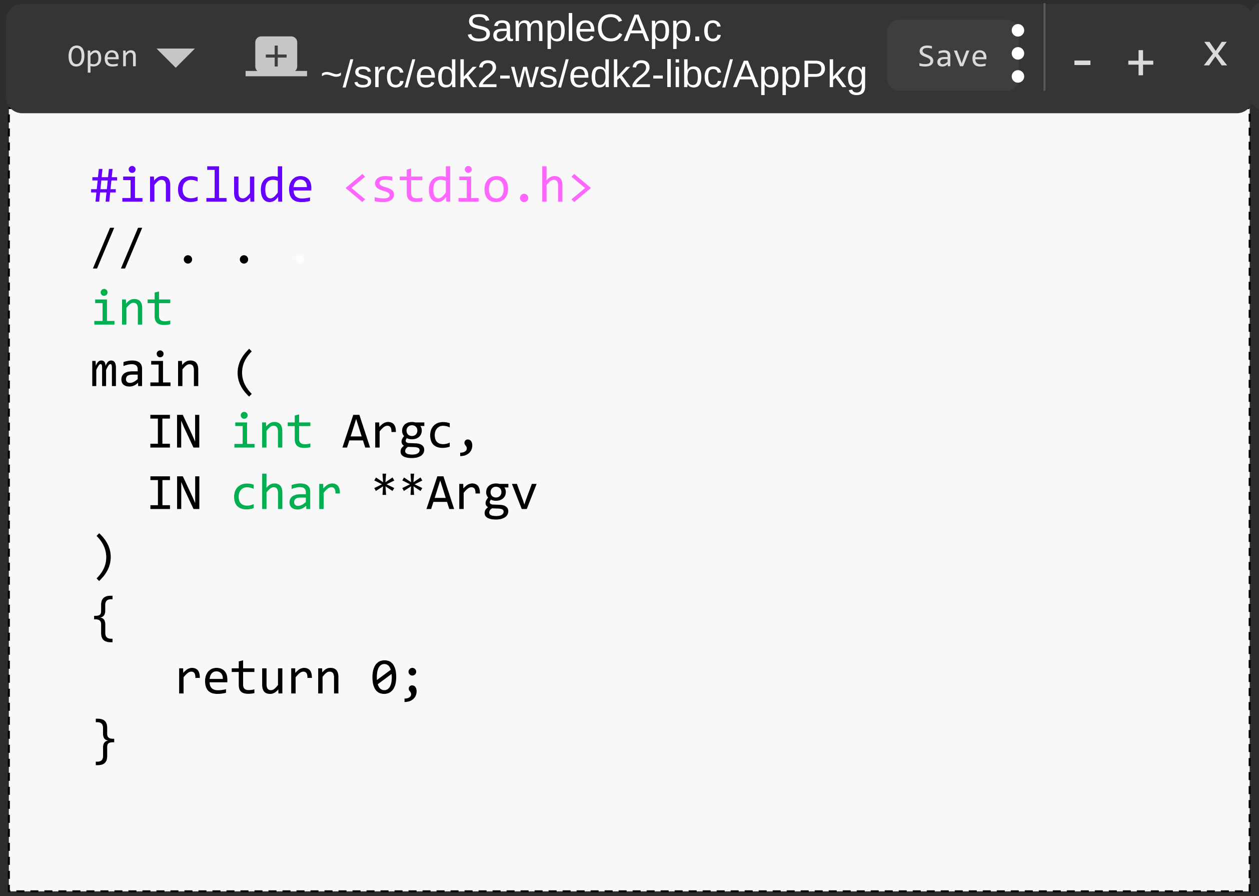
- Copy and paste directory ~../FW/LabSampleCode/SampleCApp to
  ~src/edk2-libc/AppPkg/Applications/SampleCApp



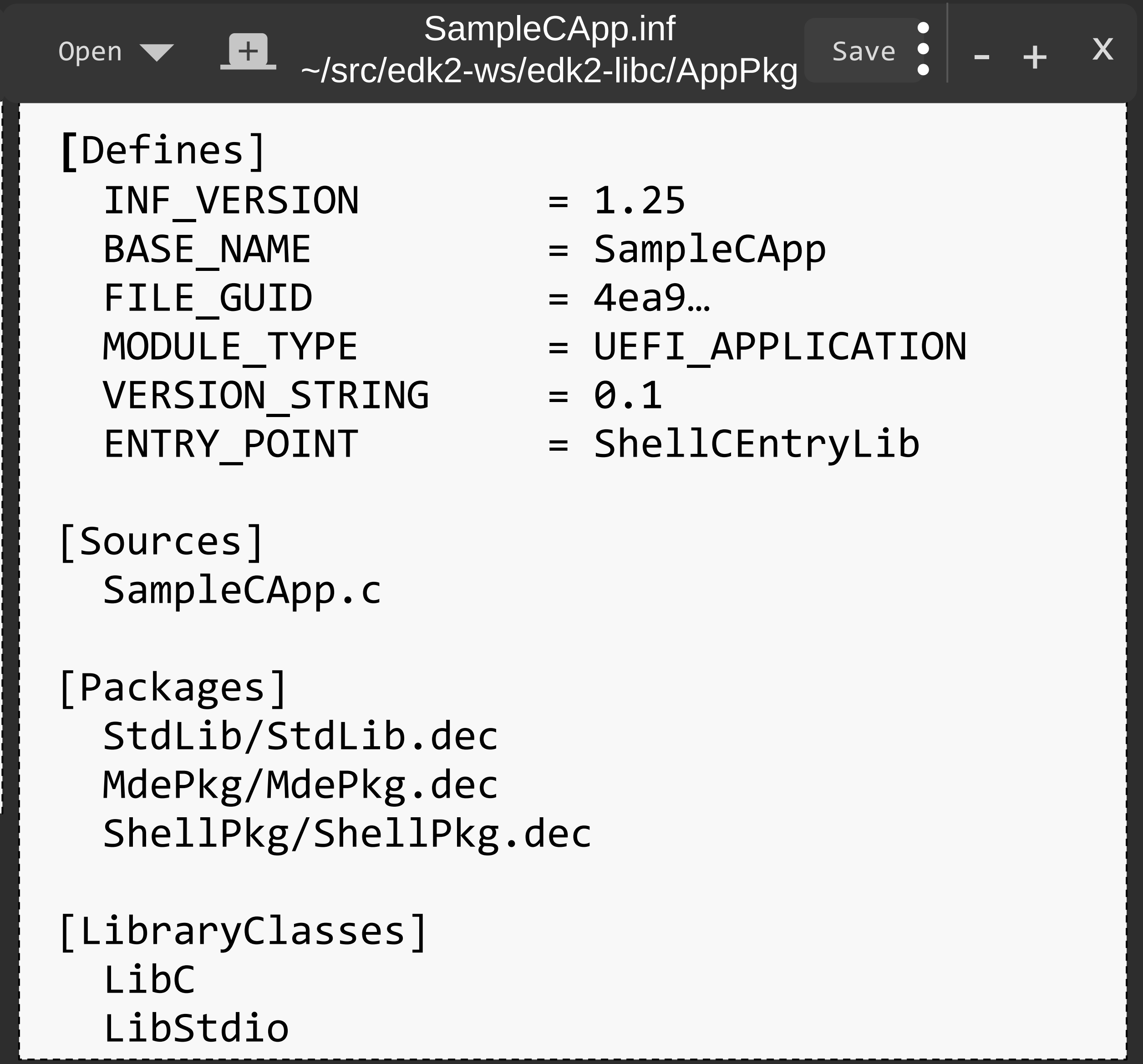
Lab 6: EDK II using EADK

Check out AppPkg/Applications/SampleCApp

SampleCApp.c and SampleCApp.inf



```c
#include <stdio.h>
// . . .
int
main (
  IN int Argc,
  IN char **Argv
)
{
   return 0;
}
```



```
[Defines]
  INF_VERSION           = 1.25
  BASE_NAME             = SampleCApp
  FILE_GUID             = 4ea9…
  MODULE_TYPE           = UEFI_APPLICATION
  VERSION_STRING        = 0.1
  ENTRY_POINT           = ShellCEntryLib

[Sources]
  SampleCApp.c

[Packages]
  StdLib/StdLib.dec
  MdePkg/MdePkg.dec
  ShellPkg/ShellPkg.dec

[LibraryClasses]
  LibC
  LibStdio
```

tianocore

Lab 6 : Update AppPkg.dsc

Edit the `~src/edk2-ws/edk-libc/AppPkg/AppPkg.dsc` and add SampleCApp.inf at the end

of the components section

- (hint: search for "#### Sample Applications")
- AppPkg/Applications/SampleCApp/SampleCApp.inf

```
[Components]
#### Sample Applications.
AppPkg/Applications/Hello/Hello.inf            # No LibC includes or functions.
AppPkg/Applications/Main/Main.inf              # Simple invocation. No other LibC function
AppPkg/Applications/Enquire/Enquire.inf        #
AppPkg/Applications/ArithChk/ArithChk.inf      #

AppPkg/Applications/SampleCApp/SampleCApp.inf  # LAB 6
```

tianocore

Lab 6 :Build and Test SampleCApp

Build the AppPkg

```
bash$ build -p AppPkg/AppPkg.dsc –m AppPkg/Applications/SampleCApp/SampleCApp.inf
```

Copy the built application to the run OVMF hda-contents directory

```
bash$ cp Build/AppPkg/DEBUG_GCC5/X64/SampleCApp.efi ~/run-ovmf/hda-contents
```

Test by Invoking Qemu

```
bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```

Run the application from the New Shell

```
Shell> SampleCApp
Shell>
```

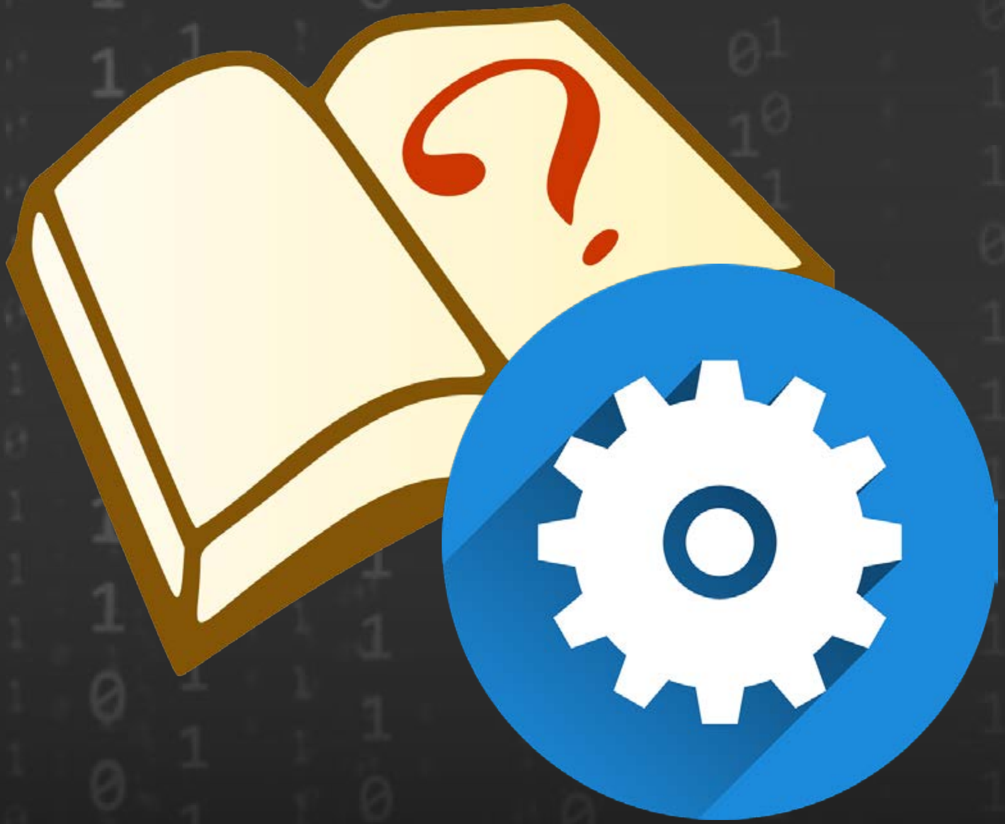
Exit QEMU

Notice that the program will immediately unload because the main function is empty

tianocore

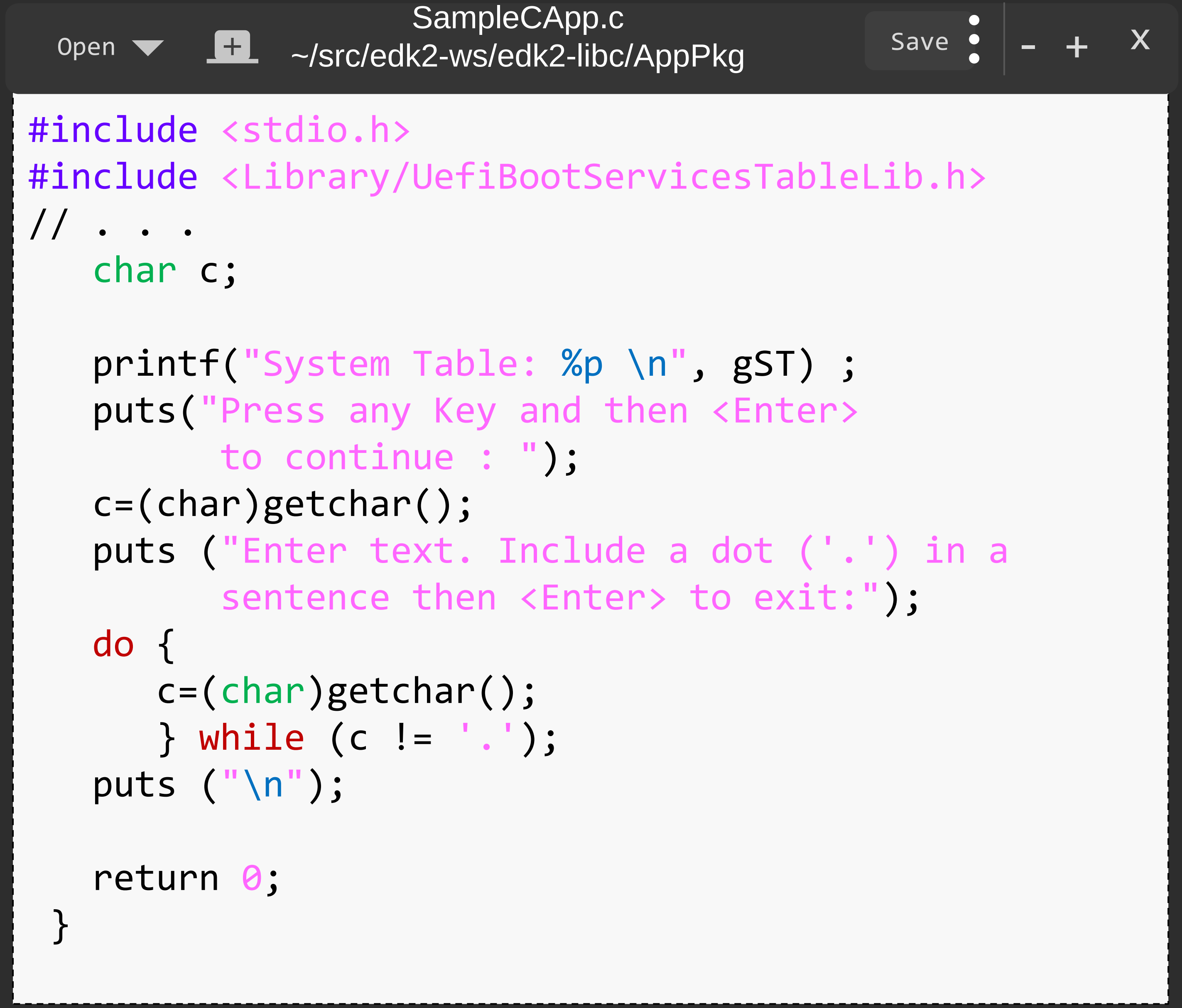
Lab 7: Adding Functionality to SampleCApp

In this lab, you'll add functionality to SampleCApp the same as in Lab 5. This lab will use EADK libraries, so the coding style is similar to standard C.

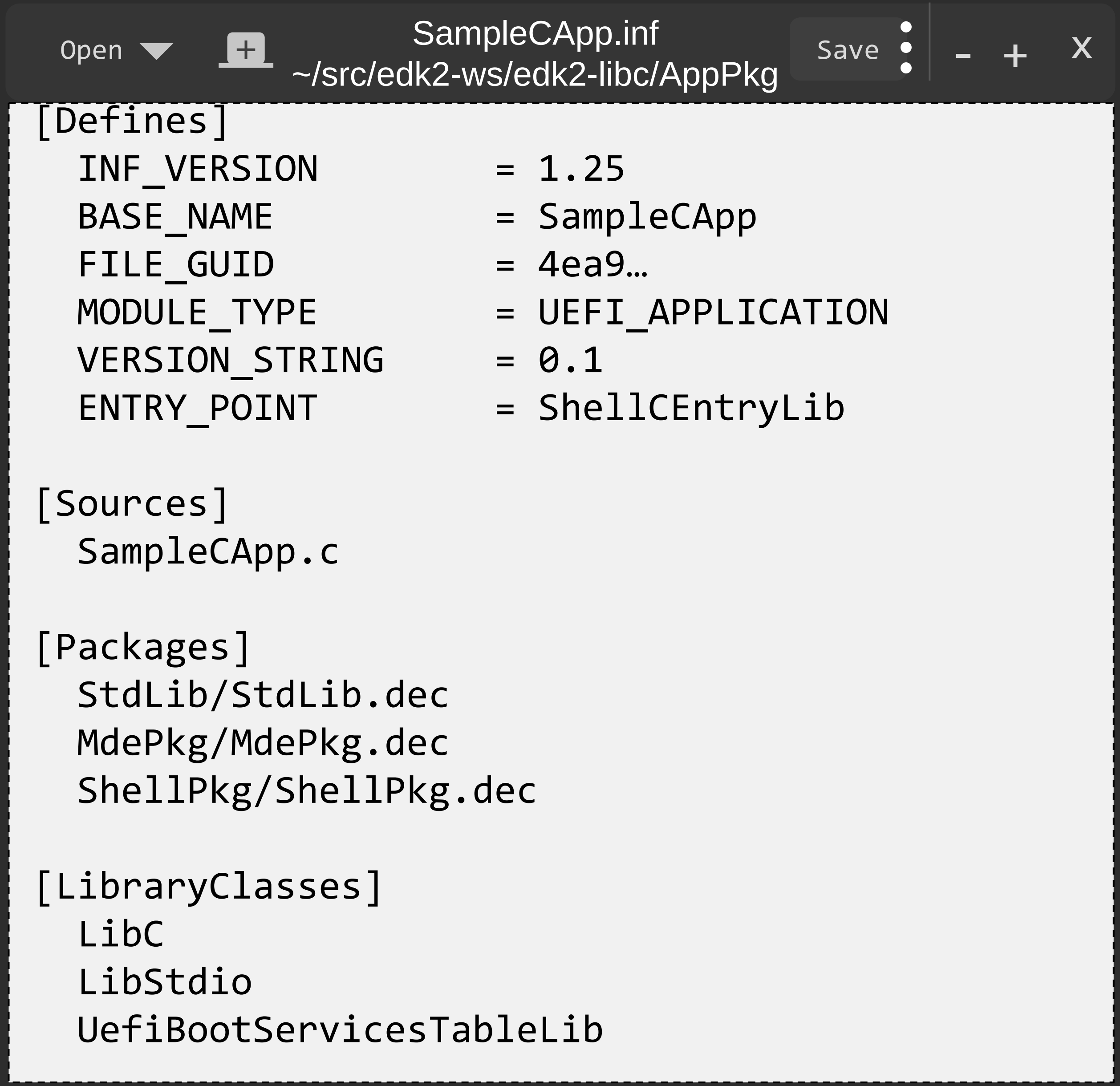


Lab 7: Add the same functionally from Lab 5

SampleCApp.c and SampleCApp.inf



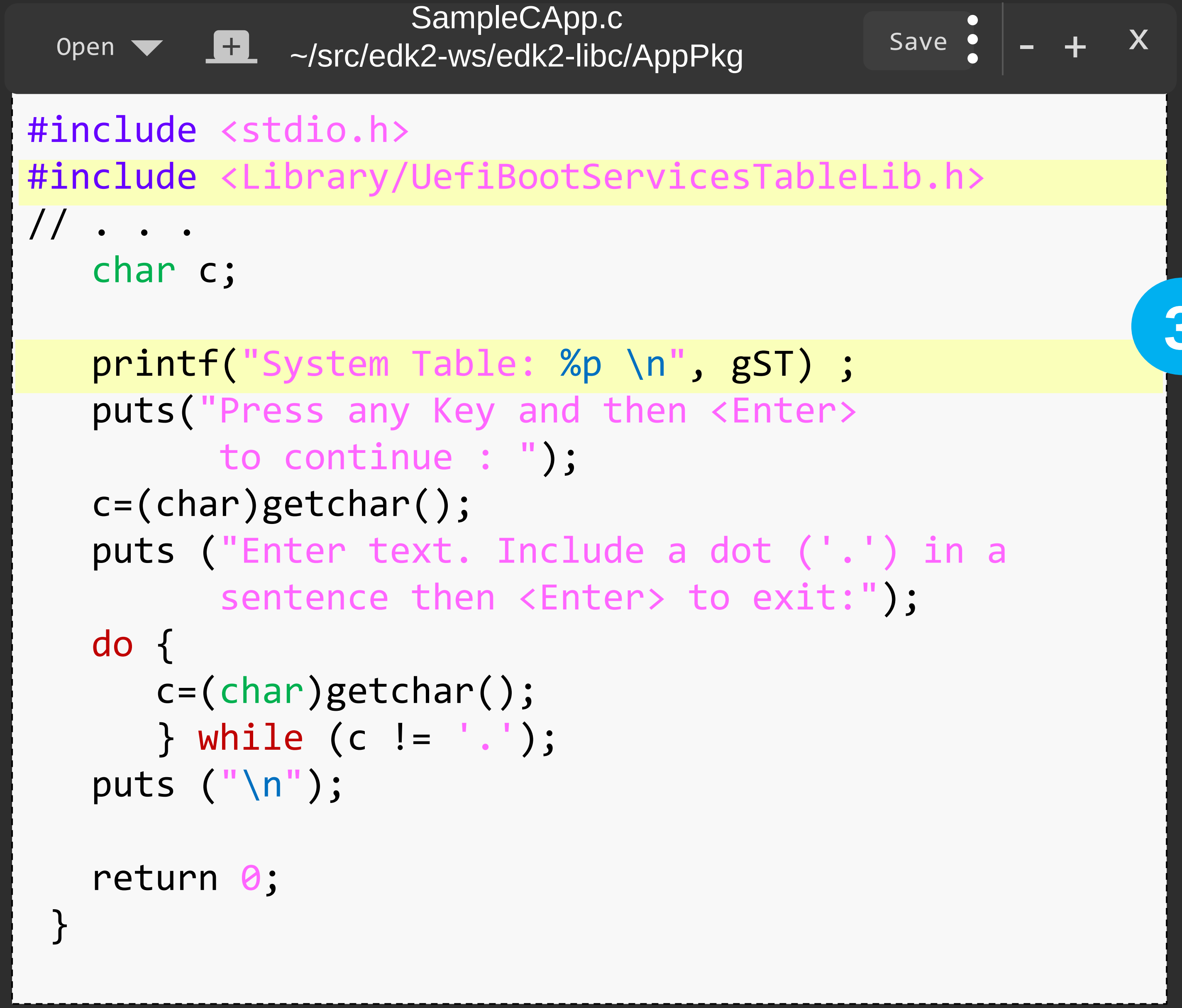
```c
#include <stdio.h>
#include <Library/UefiBootServicesTableLib.h>
// . . .
   char c;

   printf("System Table: %p \n", gST) ;
   puts("Press any Key and then <Enter>
         to continue : ");
   c=(char)getchar();
   puts ("Enter text. Include a dot ('.') in a
         sentence then <Enter> to exit:");
   do {
      c=(char)getchar();
      } while (c != '.');
   puts ("\n");

   return 0;
 }
```



```
[Defines]
  INF_VERSION         = 1.25
  BASE_NAME           = SampleCApp
  FILE_GUID           = 4ea9…
  MODULE_TYPE         = UEFI_APPLICATION
  VERSION_STRING      = 0.1
  ENTRY_POINT         = ShellCEntryLib

[Sources]
  SampleCApp.c

[Packages]
  StdLib/StdLib.dec
  MdePkg/MdePkg.dec
  ShellPkg/ShellPkg.dec

[LibraryClasses]
  LibC
  LibStdio
  UefiBootServicesTableLib
```

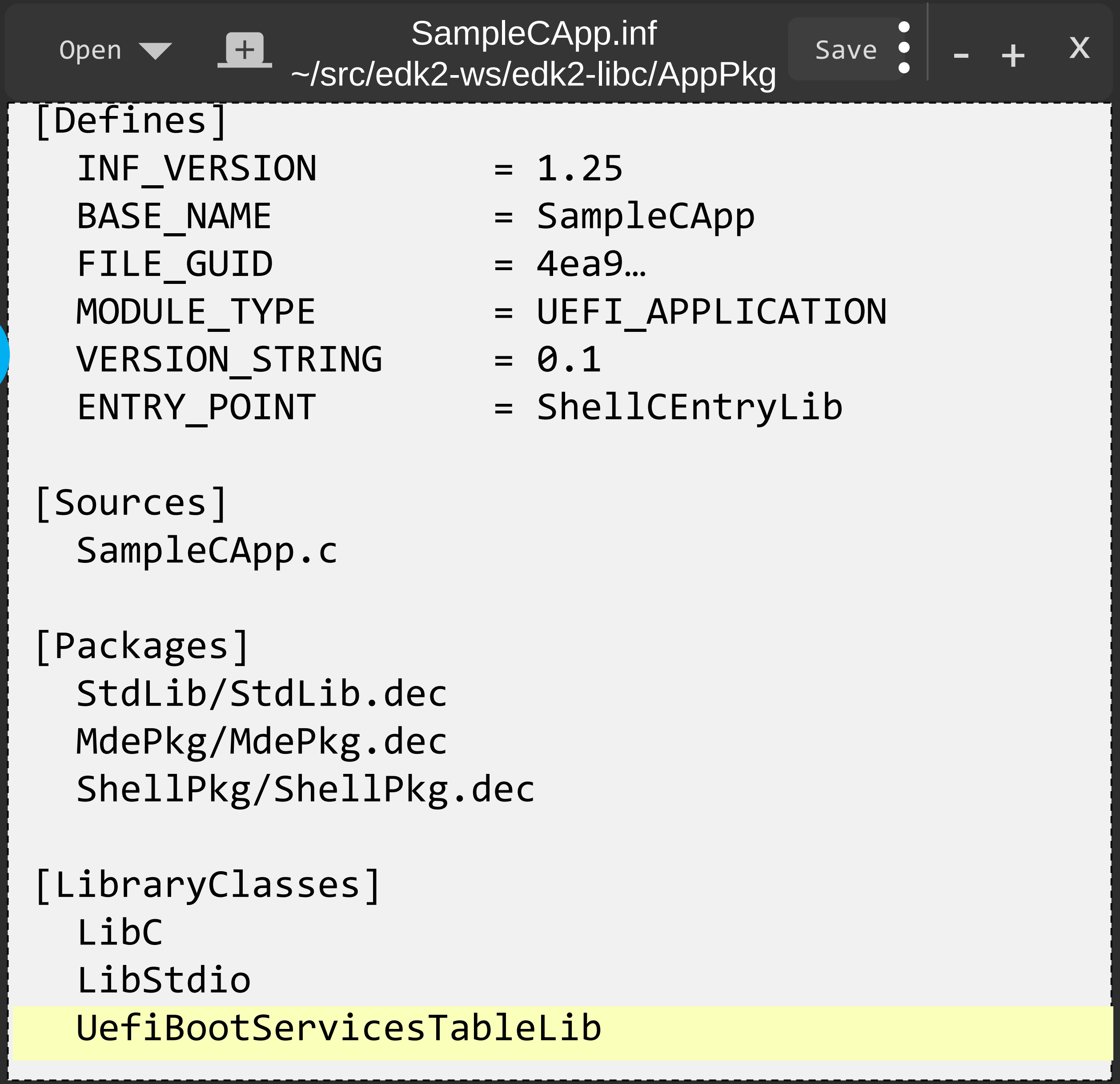
Lab 7: Add the same functionally from Lab 5

SampleCApp.c and SampleCApp.inf



```c
#include <stdio.h>
#include <Library/UefiBootServicesTableLib.h>
// . . .
   char c;

   printf("System Table: %p \n", gST) ;
   puts("Press any Key and then <Enter>
         to continue : ");
   c=(char)getchar();
   puts ("Enter text. Include a dot ('.') in a
         sentence then <Enter> to exit:");
   do {
      c=(char)getchar();
      } while (c != '.');
   puts ("\n");

   return 0;
 }
```

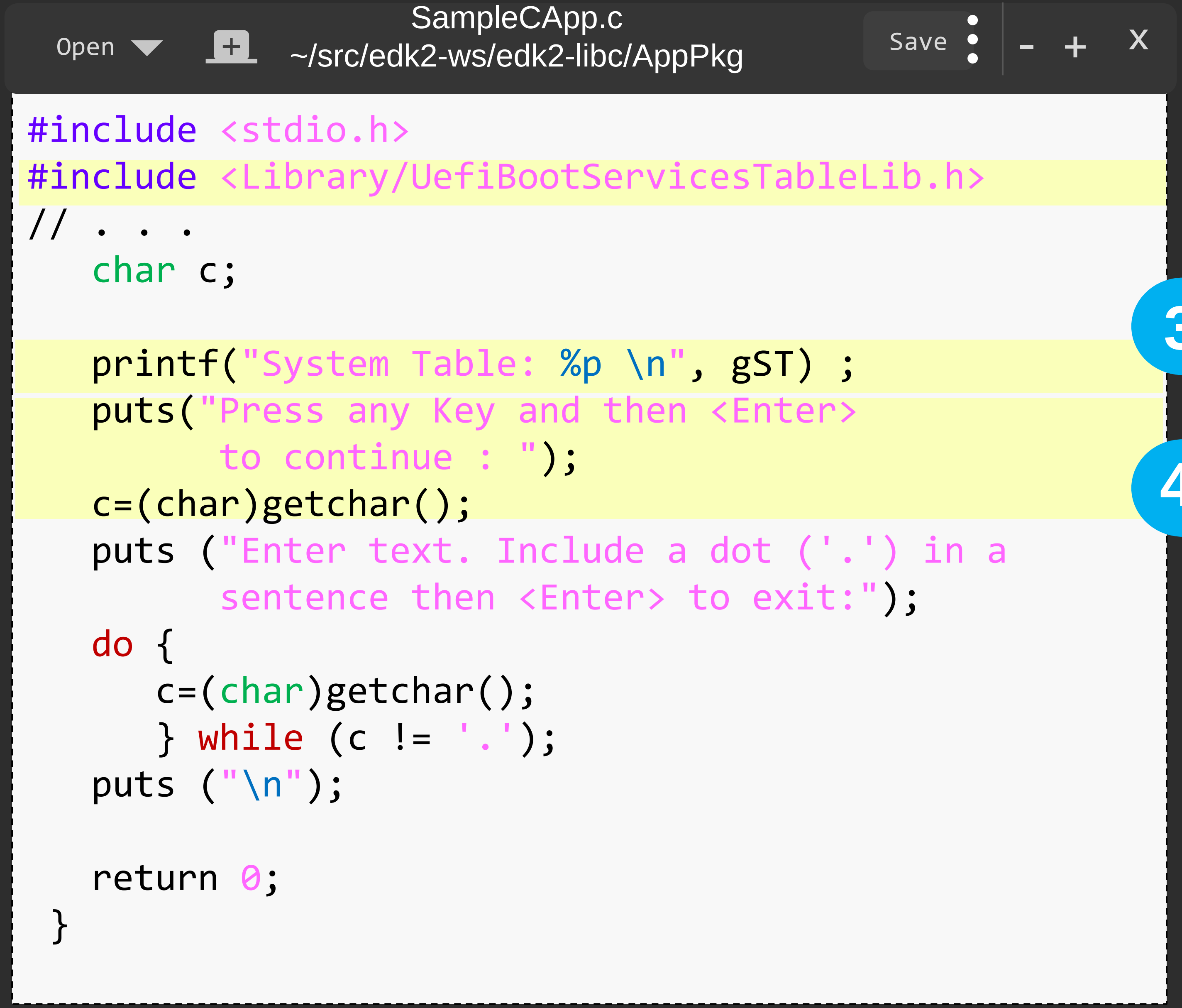
3



```
[Defines]
  INF_VERSION          = 1.25
  BASE_NAME            = SampleCApp
  FILE_GUID            = 4ea9…
  MODULE_TYPE          = UEFI_APPLICATION
  VERSION_STRING       = 0.1
  ENTRY_POINT          = ShellCEntryLib

[Sources]
  SampleCApp.c

[Packages]
  StdLib/StdLib.dec
  MdePkg/MdePkg.dec
  ShellPkg/ShellPkg.dec

[LibraryClasses]
  LibC
  LibStdio
  UefiBootServicesTableLib
```

Lab 7: Add the same functionally from Lab 5

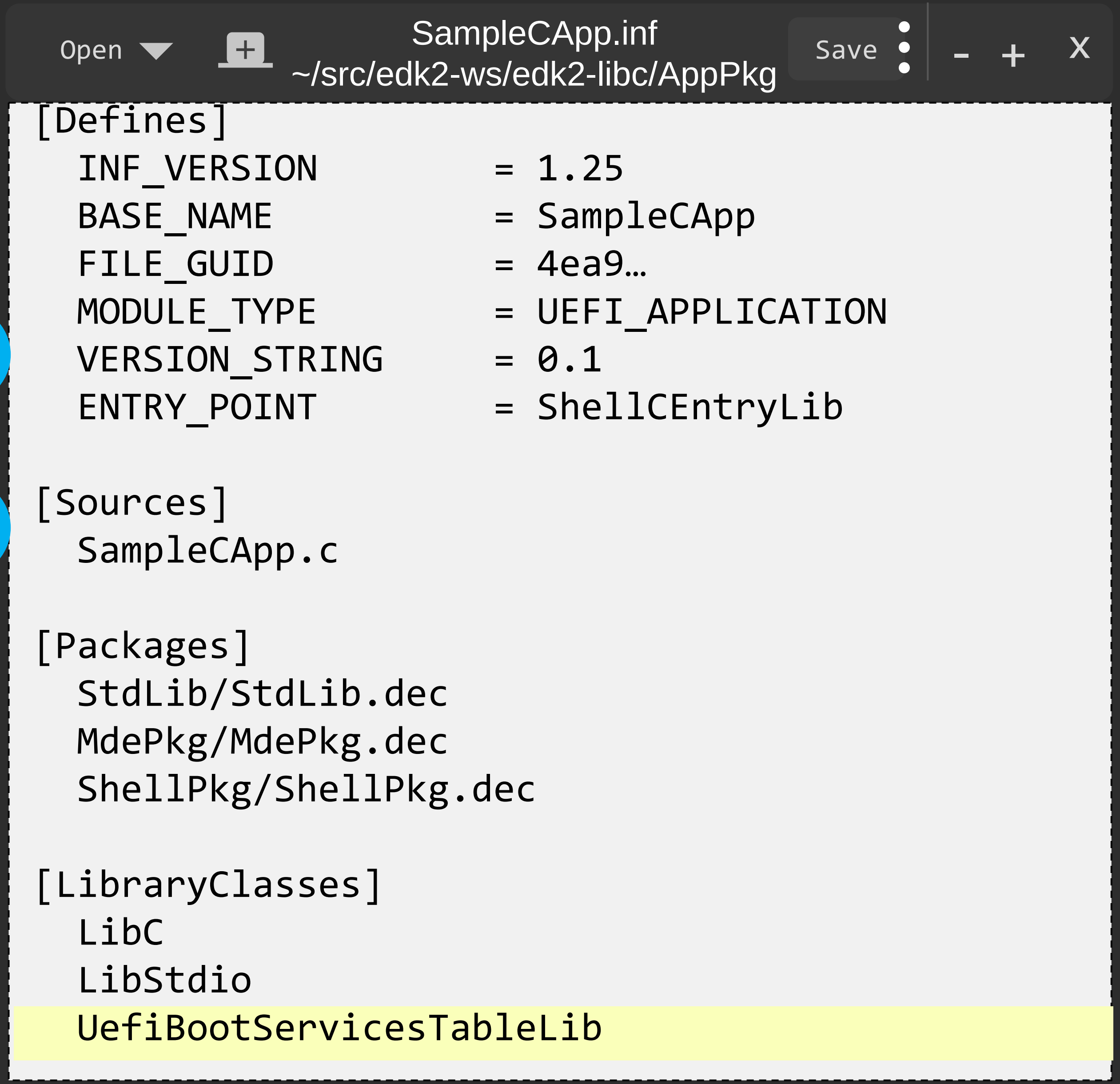
SampleCApp.c and SampleCApp.inf


Open SampleCApp.c ~/src/edk2-ws/edk2-libc/AppPkg Save - + X

```
#include <stdio.h>
#include <Library/UefiBootServicesTableLib.h>
// . . .
   char c;

   printf("System Table: %p \n", gST) ;
   puts("Press any Key and then <Enter>
         to continue : ");
   c=(char)getchar();
   puts ("Enter text. Include a dot ('.') in a
         sentence then <Enter> to exit:");
   do {
      c=(char)getchar();
      } while (c != '.');
   puts ("\n");

   return 0;
 }
```

3

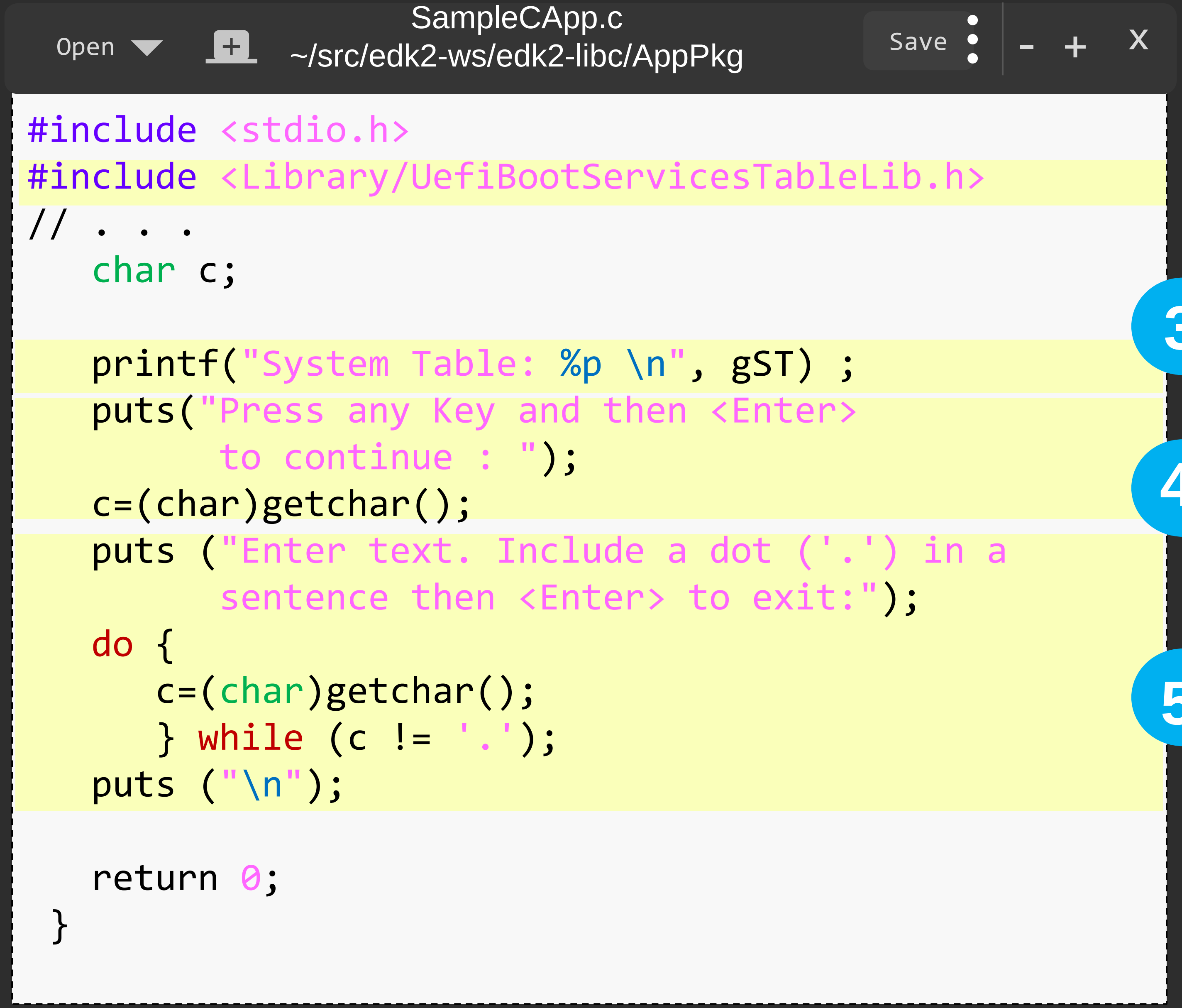
4


Open SampleCApp.inf ~/src/edk2-ws/edk2-libc/AppPkg Save - + X

```
[Defines]
  INF_VERSION             = 1.25
  BASE_NAME               = SampleCApp
  FILE_GUID               = 4ea9…
  MODULE_TYPE             = UEFI_APPLICATION
  VERSION_STRING          = 0.1
  ENTRY_POINT             = ShellCEntryLib

[Sources]
  SampleCApp.c

[Packages]
  StdLib/StdLib.dec
  MdePkg/MdePkg.dec
  ShellPkg/ShellPkg.dec

[LibraryClasses]
  LibC
  LibStdio
  UefiBootServicesTableLib
```

Lab 7: Add the same functionally from Lab 5

SampleCApp.c and SampleCApp.inf

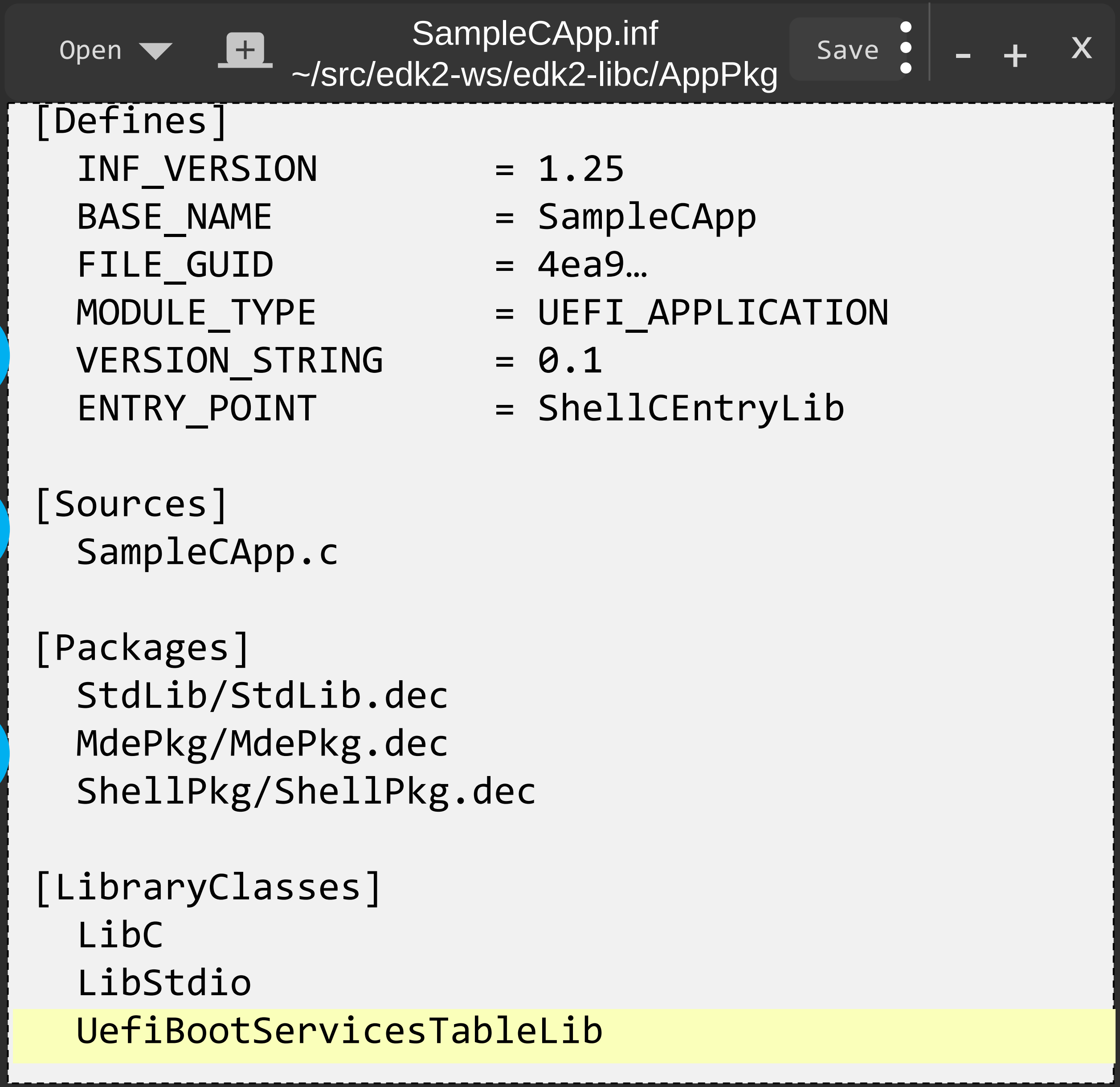


```c
#include <stdio.h>
#include <Library/UefiBootServicesTableLib.h>
// . . .
   char c;

   printf("System Table: %p \n", gST) ;
   puts("Press any Key and then <Enter>
         to continue : ");
   c=(char)getchar();
   puts ("Enter text. Include a dot ('.') in a
         sentence then <Enter> to exit:");
   do {
      c=(char)getchar();
      } while (c != '.');
   puts ("\n");

   return 0;
 }
```

3

4

5



```
[Defines]
  INF_VERSION                    = 1.25
  BASE_NAME                      = SampleCApp
  FILE_GUID                      = 4ea9…
  MODULE_TYPE                    = UEFI_APPLICATION
  VERSION_STRING                 = 0.1
  ENTRY_POINT                    = ShellCEntryLib

[Sources]
  SampleCApp.c

[Packages]
  StdLib/StdLib.dec
  MdePkg/MdePkg.dec
  ShellPkg/ShellPkg.dec

[LibraryClasses]
  LibC
  LibStdio
  UefiBootServicesTableLib
```

tianocore

Lab 7: Solution

SampleCApp.c and SampleCApp.inf

“C” file

```
#include <stdio.h>
#include <Library/UefiBootServicesTable
// . . .
   char c;

   printf("System Table: %p \n", gST) ;
   puts("Press any Key and then <Enter> to continue :
   c=(char)getchar();
   puts ("Enter text. Include a dot ('.') in a
   do {
      c=(char)getchar();
      } while (c != '.');
   puts ("\n");

   return 0;
 }
```

.inf file

```
[Defines]
  INF_VERSION    = 1.25
  BASE_NAME      = SampleCApp
  FILE_GUID      = 4ea9…
  MODULE_TYPE    = UEFI_APPLICATION
  VERSION_STRING = 0.1
  ENTRY_POINT    = ShellCEntryLib

[Sources]
  SampleCApp.c

[Packages]
  StdLib/StdLib.dec
  MdePkg/MdePkg.dec
  ShellPkg/ShellPkg.dec

[LibraryClasses]
  LibC
  LibStdio
  UefiBootServicesTableLib
```

Copy and paste LabGuide.md

tianocore

Lab 7 :Build and Test SampleCApp

Build the AppPkg

```
bash$ build -p AppPkg/AppPkg.dsc –m AppPkg/Applications/SampleCApp/SampleCApp.inf
```

Copy the built application to the run OVMF hda-contents directory

```
bash$ cp Build/AppPkg/DEBUG_GCC5/X64/SampleCApp.efi ~/run-ovmf/hda-contents
```

Test by Invoking Qemu

```
bash$ cd ~/run-ovmf
bash$ . RunQemu.sh
```

Run the application from the New Shell

```
Shell> SampleCApp
Press any Key and then <Enter> to Continue :

Enter text. Include a dot (‘.’) in a sentence then <Enter> to exit:
This is sample text.
Shell>
```

tianocore

Summary

- UEFI Application with PCDs
- Simple UEFI Application
- Add functionality to UEFI Application
- Using EADK with UEFI Application

tianocore

Questions?

Return to Main Training Page

Return to Training Table of contents for next presentation link

www.tianocore.org

tianocore

ACKNOWLEDGEMENTS