

# UEFI & EDK II Training

## How to Write a UEFI Application w/ Windows Lab

See also [LabGuide.md](#) for Copy & Paste examples in labs

[tianocore.org](https://tianocore.org)



# LESSON OBJECTIVE

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

# UEFI APPLICATION W/ PCDS



## EDK II PCD's Purpose and Goals

 Documentaton : [MdeModulePkg/Universal/PCD/Dxe/Pcd.inf](https://github.com/tianocore/edk2/blob/master/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**

**Define  
PCD**

**Package**

**.INF**

**Reference  
PCD**

**Module**

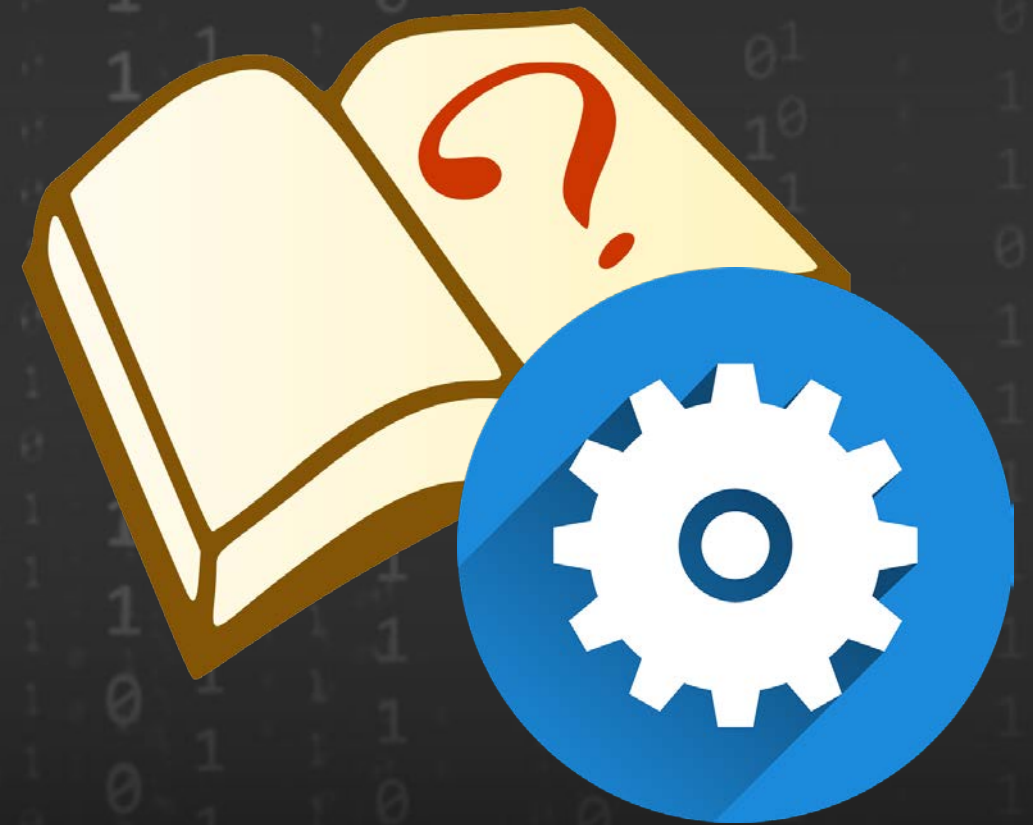
**.DSC**

**Modify  
PCD**

**Platform**

# 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

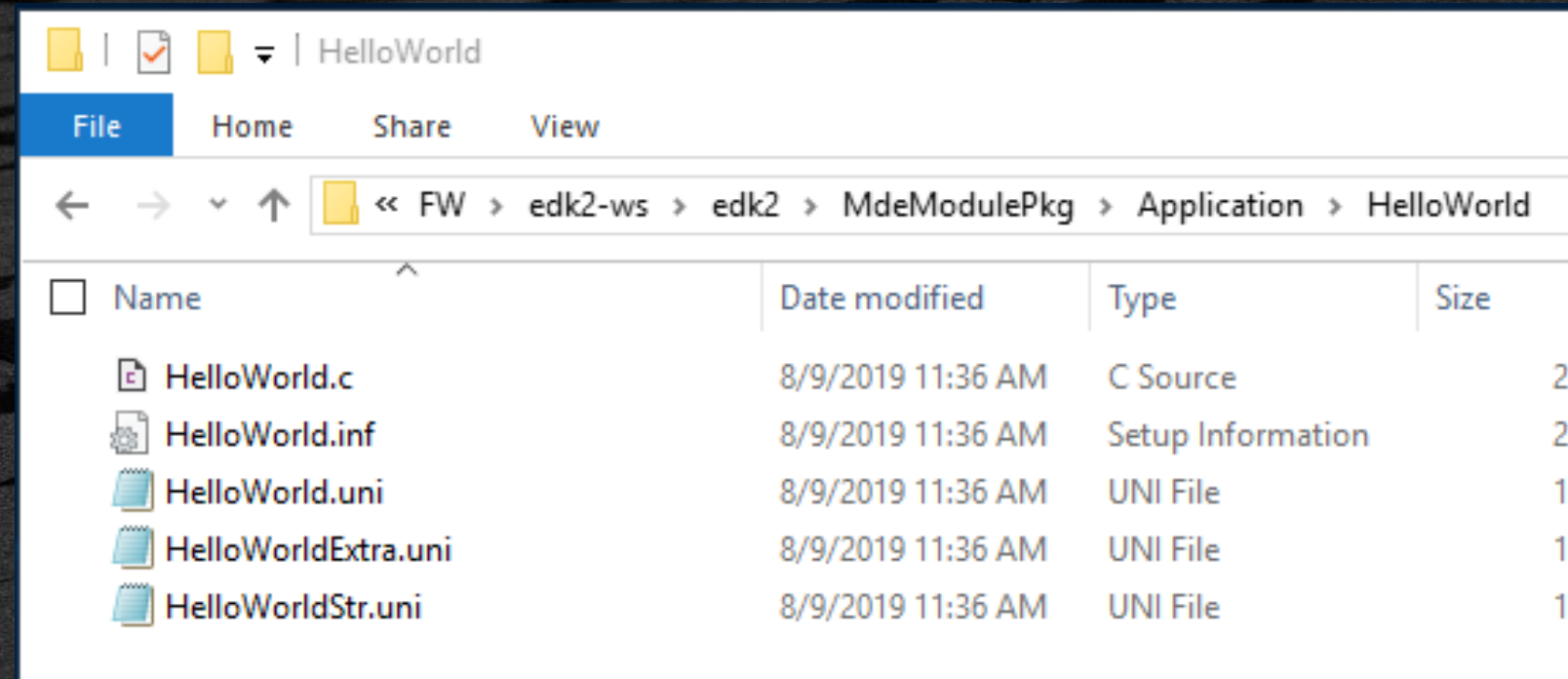
First Setup for Building EDK II for EmulatorPkg, See [Lab Setup](#)

Locate and Open edk2\MdeModulePkg\Application\HelloWorld\HelloWorld.c

Notice the PCD values

Build Emulation Package

Then Run HelloWorld





# EDK II HelloWorld App Lab

Open a VS Command Prompt and type: `cd C:/FW/edk2-ws` then

```
$> Setenv.bat  
$> cd edk2  
$> edksetup
```

Build EmulatorPkg for Windows X64 (run WinHost.exe from Build/EmulatorX64/ . . . ./X64 )

```
$> Build -D ADD_SHELL_STRING  
$> RunEmulator.bat
```

At the UEFI Shell prompt

```
Shell> HelloWorld  
UEFI Hello World!  
Shell>
```

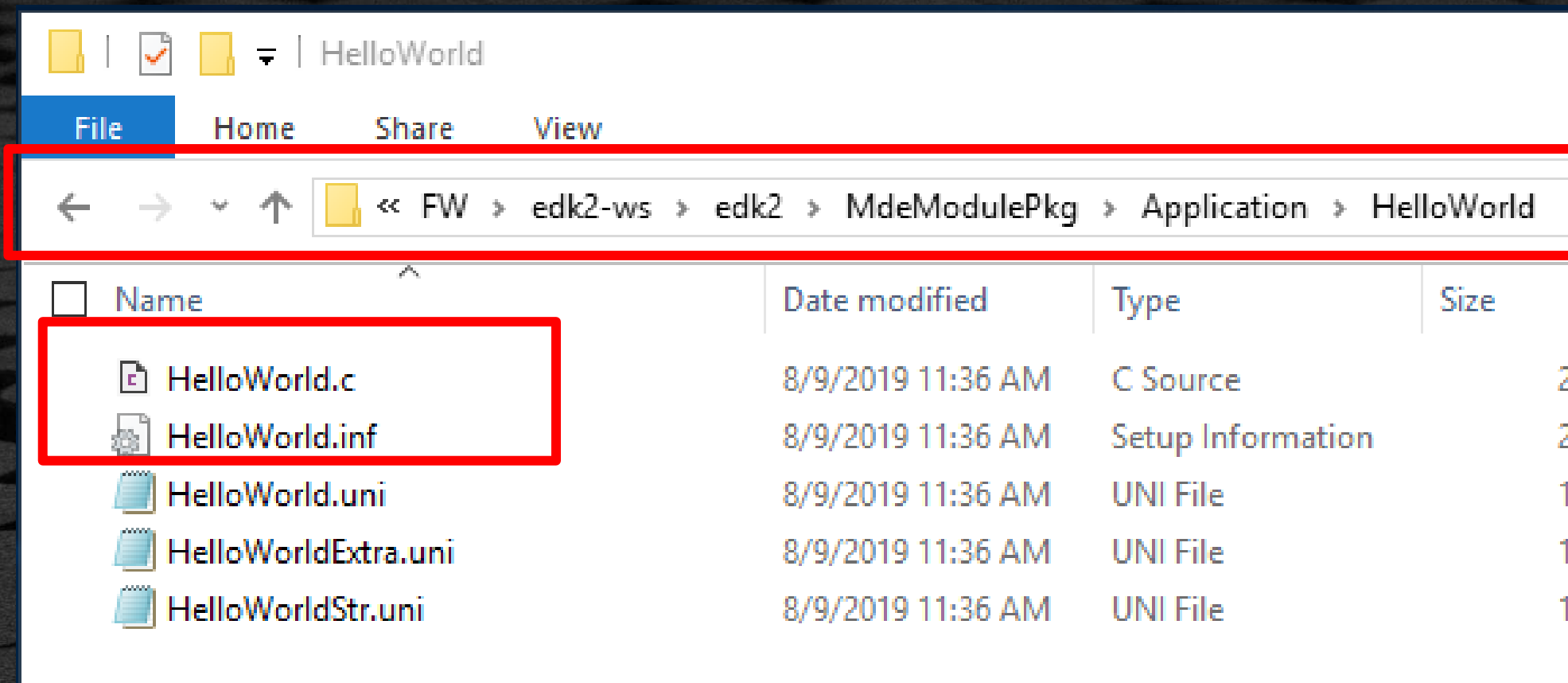
How can we force the HelloWorld application to print out 3 times ?

Note: RunEmulator.bat will run WinHost.exe from Build/EmulatorX64/DEBUG\_**TAG**/X64



# EDK II HelloWorld App Lab

 [MdeModulePkg/Application/HelloWorld](#)





```

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;
}

```

Notice the 3 PCDs



# EDK II HelloWorld App Solution

1. Edit the file C:/FW/edk2-ws/edk2/EmulatorPkg/EmulatorPkg.dsc  
After the section [PcdsFixedAtBuild] (search for "PcdsFixedAtBuild" or "Hello")

EmulatorPkg.dsc-Notepad

File Edit Format View Help

[PcdsFixedAtBuild]

gEfiMdeModulePkgTokenSpaceGuid.PcdHelloWorldPrintTimes | 3

2. Re-Build – Cd to C:/FW/edk2-ws/edk2

```
$> Build -D ADD_SHELL_STRING
```



# EDK II HelloWorld App Solution

3. Run Emulation (run WinHost.exe from Build/EmulatorX64/ . . ./X64 )

```
C:/FW/edk2-ws/edk2> RunEmulator.bat
```

4. At the Shell prompt

```
Shell> HelloWorld
UEFI Hello World!
UEFI Hello World!
UEFI Hello World!
Shell>
```

5: Exit Emulation

```
Shell> Reset
```

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

Also see ../edk2/MdeModulePkg/MdeModulePkg.Dec



## 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]

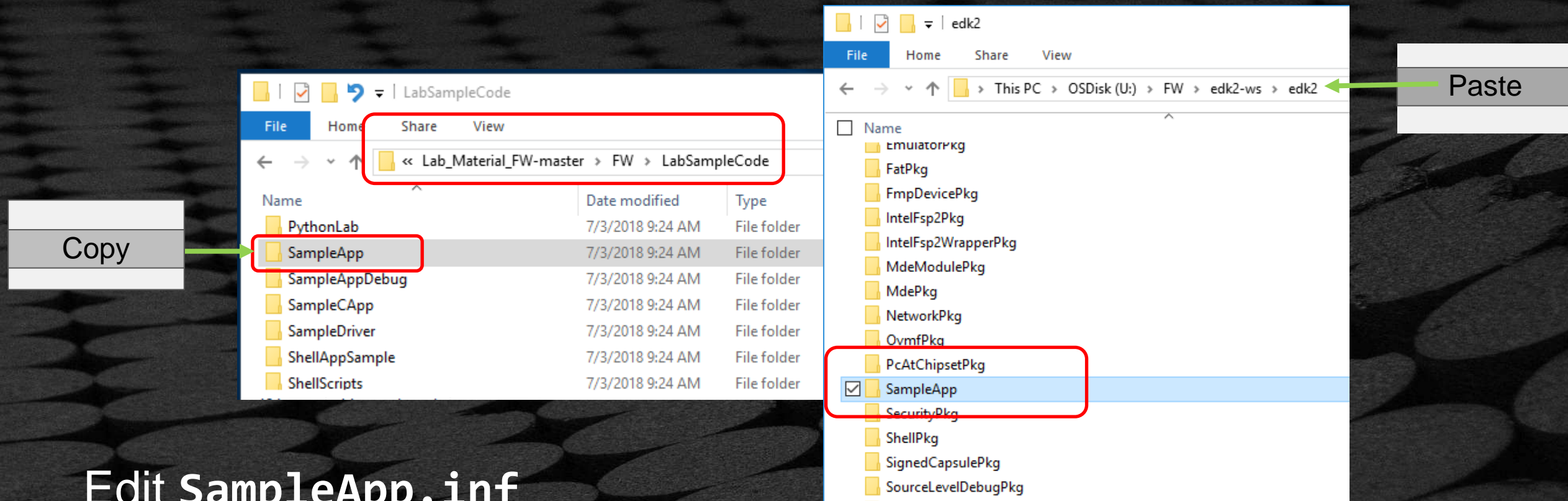
[LibraryClasses]
```

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



# Application Lab –start with .c and .inf template

Copy the LabSampleCode/SampleApp directory to C:/FW/edk2-ws/edk2



Edit SampleApp.inf

- Look in the INF for “xxxxxxxxxxxx” sections that will need information
- Create Name & GUID, and then fill in the MODULE\_TYPE



# Lab 2: Sample Application INF file

```
SampleApp.inf - Notepad
File Edit Format View Help

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

[Sources]
  XXXXXXXXXXXX

[Packages]
  #XXXXXXXXXX

[LibraryClasses]
  #XXXXXXXXXXXXXXXXXX

[Guids]
  # . . .
```

SampleApp  
Get a GUID  
UEFI\_APPLICATION

SampleApp.c

Get a GUID [guidgenerator.com/](https://guidgenerator.com/)

Copy and paste [LabGuide.md](#)



## Lab 2: Sample Application 'C' file

```
SampleApp.c - Notepad
File Edit Format View Help

/** @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



# Application Lab – Update Files

## 1. **.DSC** (EmulatorPkg/EmulatorPkg.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)

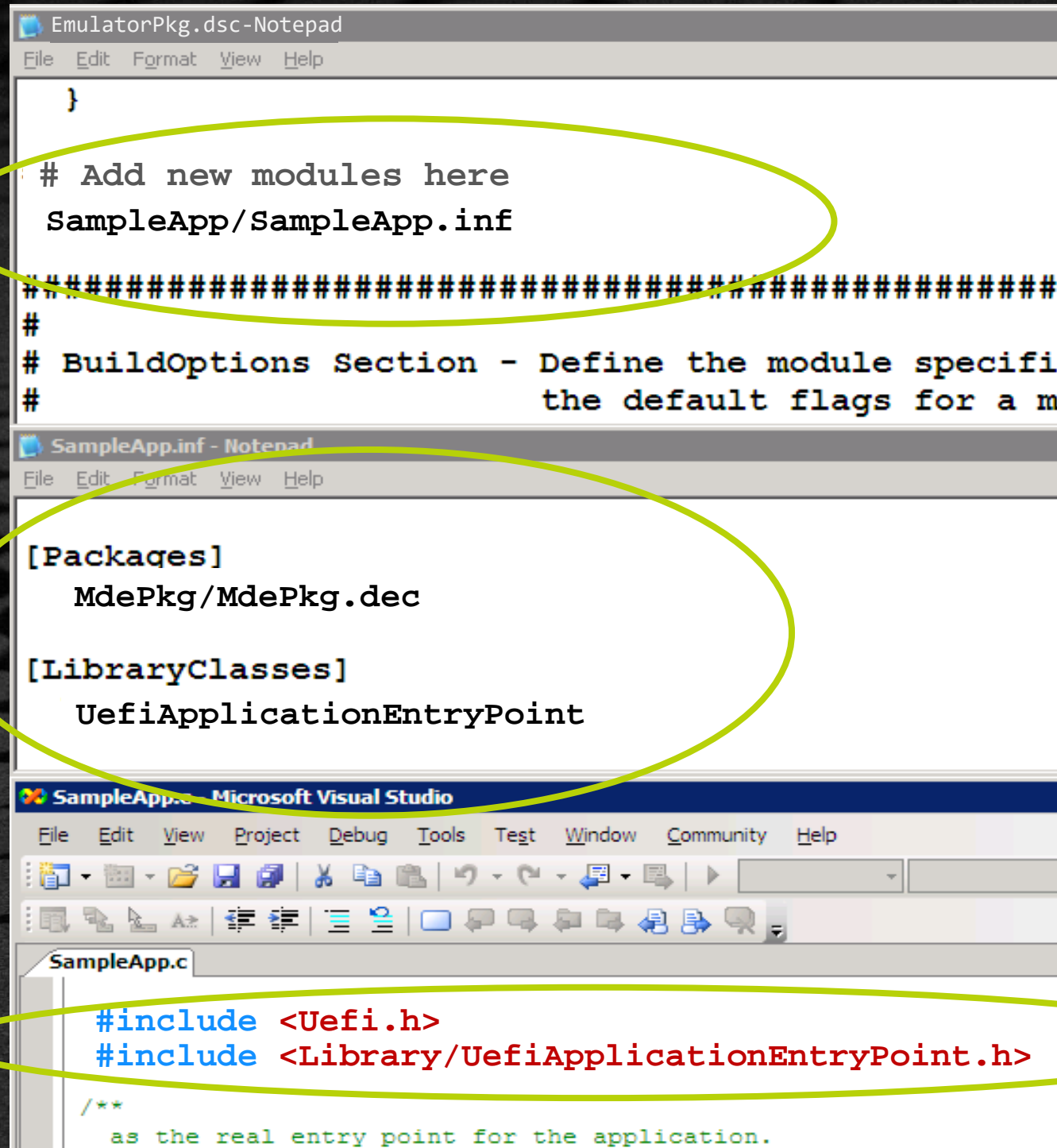
```
#include <Uefi.h>
```

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

Copy and paste [LabGuide.md](#)



# Lab 2: cont. Solution



```
EmulatorPkg.dsc-Notepad
File Edit Format View Help

}

# Add new modules here
SampleApp/SampleApp.inf

#####

#
# BuildOptions Section - Define the module specific
# the default flags for a m

SampleApp.inf - Notepad
File Edit Format View Help

[Packages]
  MdePkg/MdePkg.dec

[LibraryClasses]
  UefiApplicationEntryPoint

SampleApp.c - Microsoft Visual Studio
File Edit View Project Debug Tools Test Window Community Help

SampleApp.c

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

/**
  as the real entry point for the application.
```

EmulatorPkg/EmulatorPkg.dsc

SampleApp/SampleApp.inf

SampleApp/SampleApp.c

Copy and paste [LabGuide.md](#)



## Lab 2 : Will it compile now?

At the VS Command Prompt

```
$> Build -D ADD_SHELL_STRING  
$> RunEmulator.bat
```

Run the application from the shell

```
Shell> SampleApp  
Shell>
```

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

Exit

```
Shell> Reset
```



# Possible Build Errors

## Error on SampleApp.inf

```
C:\ Developer Command Prompt for VS2015
Processing meta-data ..
build...
c:\fw\edk2\SampleApp\SampleApp.inf(21): error 3000: No value specified
FILE_GUID =
- Failed -
Build end time: 09:11:30, Jul.25 2018
Build total time: 00:00:03
C:\fw\edk2>
```

```
C:\ Developer Command Prompt for VS2015
Processing meta-data .....
build...
: error C0DE: Unknown fatal error when processing [c:\fw\edk2\SampleApp\SampleApp.inf]
(Please send email to edk2-devel@lists.01.org for help, attaching following call stack trace!)
(Python 2.7.14 on win32) Traceback (most recent call last):
  File "build\build.py", line 2493, in Main
  File "build\build.py", line 2226, in Launch
  File "build\build.py", line 2047, in _MultiThreadBuildPlatform
  File "c:\Users\Public\Documents\BuildPool\BaseTools\build\Source\Python\AutoGen\AutoGen.py",
line 4391, in CreateCodeFile
  File "c:\Users\Public\Documents\BuildPool\BaseTools\build\Source\Python\AutoGen\AutoGen.py",
line 3604, in _GetAutoGenFileList
  File "c:\Users\Public\Documents\BuildPool\BaseTools\build\Source\Python\AutoGen\GenC.py", lin
e 2075, in CreateCode
  File "c:\Users\Public\Documents\BuildPool\BaseTools\build\Source\Python\AutoGen\GenC.py", lin
e 2033, in CreateHeaderCode
  File "c:\Users\Public\Documents\BuildPool\BaseTools\build\Source\Python\Common\Misc.py", line
308, in GuidStringToGuidStructureString
IndexError: list index out of range
- Failed -
Build end time: 09:15:55, Jul.25 2018
Build total time: 00:00:24
```

The FILE\_GUID was invalid or not updated from “XXX...” to a proper formatted GUID



# Possible Build Errors

## Error on SampleApp.inf

```
c:\ Developer Command Prompt for VS2015
Building ... c:\fw\edk2\MdeModulePkg\Universal\LoadFileOnFv2\LoadFileOnFv2.inf [IA32]
SampleApp.c
  B Creating library c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SecMain.lib and object c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SecMain.exp
Building ... c:\fw\edk2\MdeModulePkg\Universal\PlatformDriOverrideDxe\PlatformDriOverrideDxe.inf [IA32]
Generating code
Building ... c:\fw\edk2\MdeModulePkg\Application\VariableInfo\VariableInfo.inf [IA32]
c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp\DEBUG\AutoGen.h(16) : fatal error C1083: Cannot open include file: 'Base.h': No such file or directory
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\bin\cl.exe"' : return code '0x2'
Stop.

build...
: error 7000: Failed to execute command
C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\bin\nmake.exe /nologo tbuild [c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp]

build...
: error F002: Failed to build module
c:\fw\edk2\SampleApp\SampleApp.inf [IA32, VS2013x86, DEBUG]

- Failed -
Build end time: 09:23:56, Jul.25 2018
Build total time: 00:00:41
```

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



# Possible Build Errors

## Compiler Error on SampleApp.c

```
c:\ Developer Command Prompt for VS2015
2\SampleApp\SampleApp\DEBUG /Ic:\fw\edk2\MdePkg /Ic:\fw\edk2\MdePkg\Include /Ic:\fw\edk
2\MdePkg\Include\Ia32 c:\fw\edk2\SampleApp\SampleApp.c
cl : Command line warning D9025 : overriding '/O1' with '/Od'
SampleApp.c
Building ... c:\fw\edk2\MdeModulePkg\Application\BootManagerMenuApp\BootManagerMenuApp.inf
[IA32]
c:\fw\edk2\SampleApp\SampleApp.c(16) : fatal error C1083: Cannot open include file: 'Libra
ry/UefiAplicationEntryPoint.h': No such file or directory
Building ... c:\fw\edk2\MdeModulePkg\Universal\LoadFileOnFv2\LoadFileOnFv2.inf [IA32]
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Visual Studio 12.0\vc\bin\cl
.exe"' : return code '0x2'
Stop.
Building ... c:\fw\edk2\MdeModulePkg\Universal\PlatformDriOverrideDxe\PlatformDriOverrideD
xe.inf [IA32]

build...
: error 7000: Failed to execute command
C:\Program Files (x86)\Microsoft Visual Studio 12.0\vc\bin\nmake.exe /nologo tbuil
d [c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp]

build...
: error F002: Failed to build module
c:\fw\edk2\SampleApp\SampleApp.inf [IA32, VS2013x86, DEBUG]
```

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



# Possible Build Errors

## Compile Linker Error on unresolved reference

```
c:\ Developer Command Prompt for VS2015
"C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\bin\link.exe" /OUT:c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp\DEBUG\SampleApp.dll /NOLOGO /NODEFAULTLIB /IGNORE:4001 /OPT:REF /OPT:ICF=10 /MAP /ALIGN:32 /SECTION:.xdata,D /SECTION:.pdata,D /MACHINE:X86 /LTCG /DLL /ENTRY:_ModuleEntryPoint /SUBSYSTEM:EFI_BOOT_SERVICE_DRIVER /SAFESEH:NO /BASE:0 /DRIVER /DEBUG /EXPORT InitializeDriver=_ModuleEntryPoint /BASE:0x10000 /ALIGN:4096 /FILEALIGN:4096 /SUBSYSTEM:CONSOLE @c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp\OUTPUT\static_library_files.lst
Building ... c:\fw\edk2\MdeModulePkg\Universal\SetupBrowserDxe\SetupBrowserDxe.inf [IA32]
LINK : error LNK2001: unresolved external symbol _ModuleEntryPoint
c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp\DEBUG\SampleApp.lib : fatal error LNK1120: 1 unresolved externals
Building ... c:\fw\edk2\MdeModulePkg\Universal\DisplayEngineDxe\DisplayEngineDxe.inf [IA32]
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\bin\link.exe"' : return code '0x400'
Stop.

build...
: error 7000: Failed to execute command
C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\bin\nmake.exe /nologo tbuild [c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp]

build...
: error F002: Failed to build module
c:\fw\edk2\SampleApp\SampleApp.inf [IA32, VS2013x86, DEBUG]
```

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



# 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



## Lab 2.1: Build Switches

In this lab, you'll remove the build switches to be always TRUE






# Build MACRO Switches

The build for EmulatorPkg 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)



```
EmulatorPkg.dsc-Notepad
File Edit Format View Help

# 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
```

First delete directory Build/EmulatorX64/DEBUG\_*tag*/X64/**ShellPkg**



# Lab 2.1: Compiling w/ Build Switch

At the VS Command Prompt build **without** the “-D ADD\_SHELL\_STRING” switch

Delete Build/EmulatorX64/DEBUG\_*tag*/X64/ShellPkg

```
$> Build
```

```
$> RunEmulator.bat
```

Check the Shell version with “Ver” command

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

Build with the -D ADD\_SHELL\_STRING

Delete Build/EmulatorX64/DEBUG\_*tag*/X64/ShellPkg

```
$> Build -D ADD_SHELL_STRING
```

```
$> RunEmulator.bat
```

Check the Shell version with  
“Ver” command

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

**NOTE:** You may need to Delete directory: Build/EmulatorX64/DEBUG\_*tag*/X64/ShellPkg  
Between each build



## Lab 2.1: Compiling w/out Build Switch

Edit the file C:/FW/edk2-ws/edk2/EmulatorPkg/EmulatorPkg.dsc

Change:

```
DEFINE ADD_SHELL_STRING = TRUE
```

(save the file)

Delete directory Build\... \Shellpkg

Re-Build – Cd to C:/FW/edk2-ws/edk2

```
$> Build
```

```
$> RunEmulator.bat
```

```
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

Copy and paste [LabGuide.md](#)



# What we learned 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/dePkg.dec
6. When using a Build Switch (-D) on the command line it overrides the value in the .DSC file



## Lab 2: If there are build errors ...

See class files for the solution

- . . .FW/LabSampleCode/LabSolutions/LessonB.2
- Copy the .inf and .c files to C:/FW/edk2-ws/edk2/SampleApp
- Search sample DSC for reference to SampleApp.inf and add this line to your workspace DSC file  
C:/FW/edk2-ws/edk2/EmulatorPkg/EmulatorPkg.dsc

SampleApp/SampleApp.inf

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

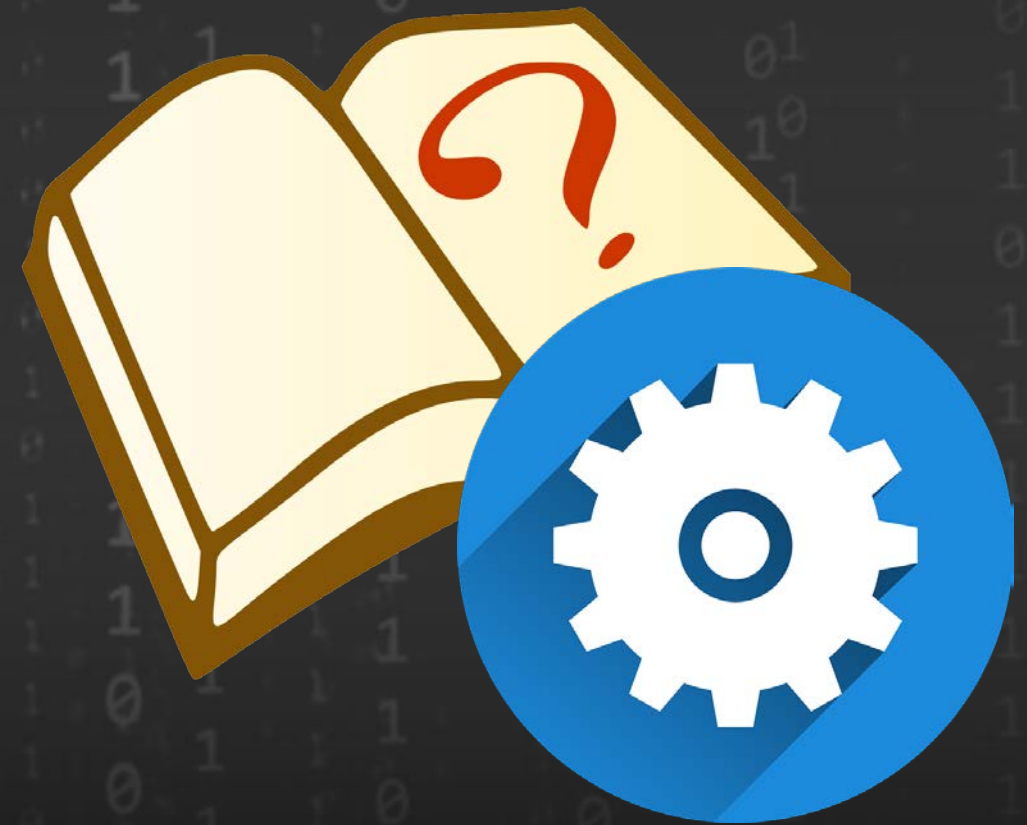
**Lab 5:** Create a Simple Typewriter function

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



## Lab 3: Print the UEFI System Table

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



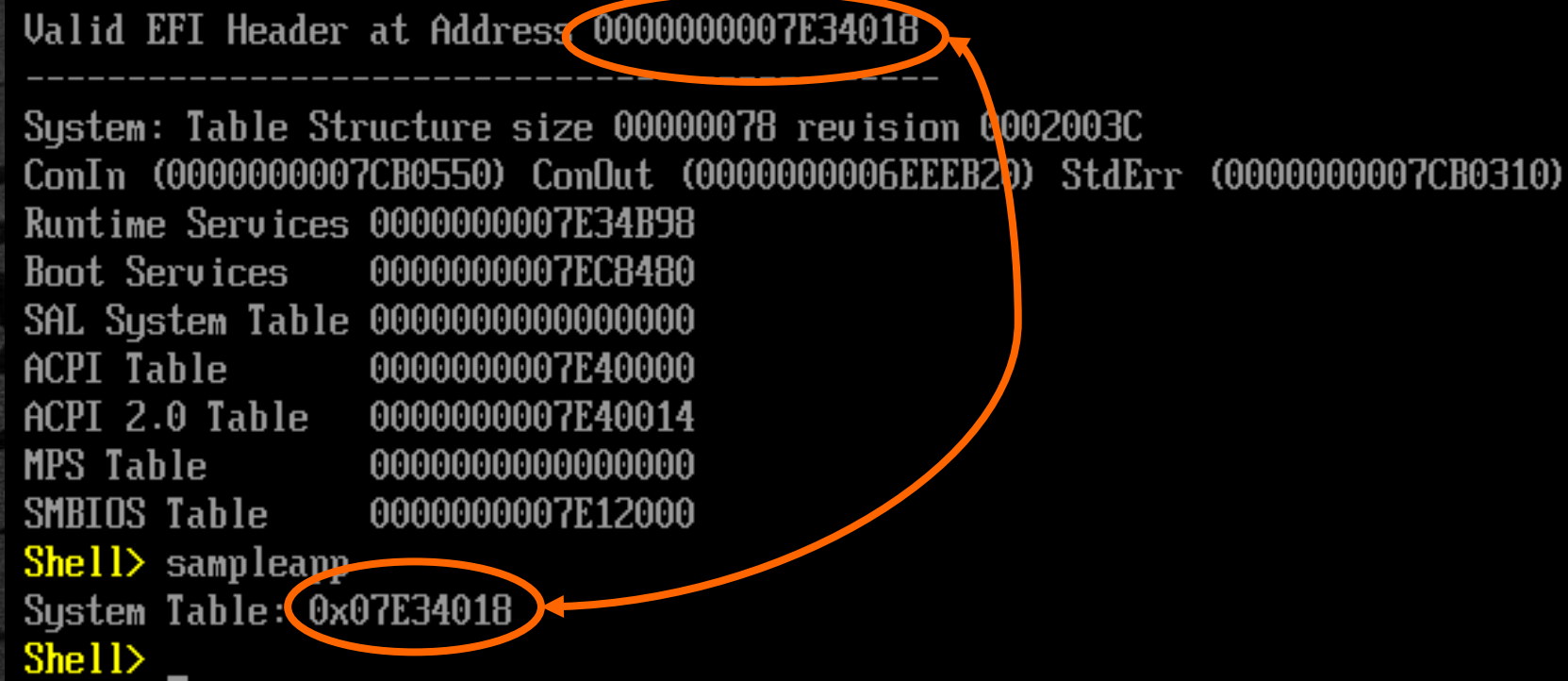


# Lab 3 : Add System Table Code

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)

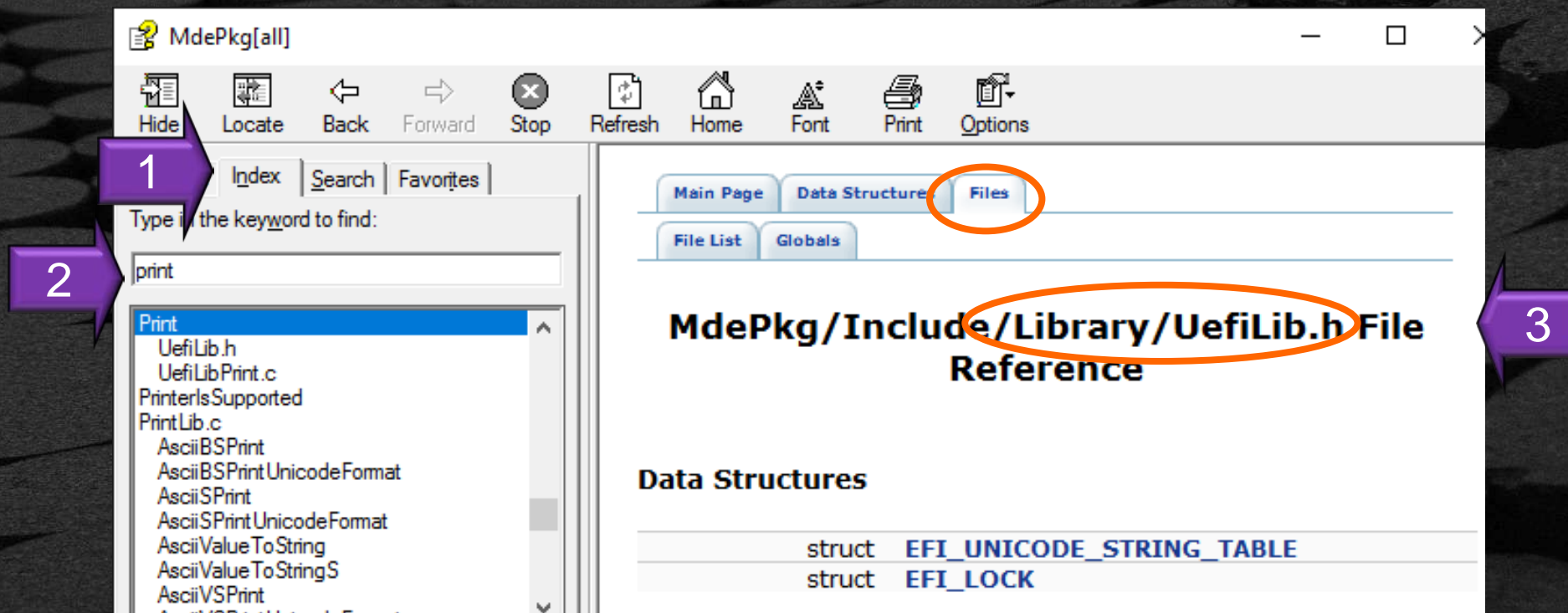
```
Valid EFI Header at Address 0000000007E34018
-----
System: Table Structure size 00000078 revision 0002003C
ConIn (0000000007CB0550) ConOut (0000000006EEEB20) StdErr (0000000007CB0310)
Runtime Services 0000000007E34B98
Boot Services    0000000007EC8480
SAL System Table 0000000000000000
ACPI Table       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





# Lab 3 : Modifying .C & .INF Files

```
SampleApp.c - Notepad
File Edit Format View Help

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 - Notepad
File Edit Format View Help

SampleApp.inf
[LibraryClasses]
    UefiApplicationEntryPoint
    UefiLib
```

Note: Solution files are in the lab materials directory

Copy and paste [LabGuide.md](#)



# Lab 3 : Build and Test SampleApp

At the VS Command Prompt

```
$> Build  
$> RunEmulator.bat
```

Run the application from the shell

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

Verify by using the “mem” command

Exit

```
Shell> Reset
```



## 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





## 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?



## 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](#) ~ In 569:



# Lab 4 : Update the C File for WaitForKey

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

```

Console.c - Notepad
File Edit Format View Help

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 - Notepad
File Edit Format View Help

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



## Lab 4: Test Compile

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

```
Building ... c:\fw\edk2\MdeModulePkg\Application\BootManagerMenuApp\BootManagerMenuApp.inf [IA32]
c:\fw\edk2\SampleApp\SampleApp.c(42) : error C2065: 'gBS' : undeclared identifier
c:\fw\edk2\SampleApp\SampleApp.c(42) : error C2223: left of '->WaitForEvent' must point to struct/union
c:\fw\edk2\SampleApp\SampleApp.c(42) : error C2065: 'gST' : undeclared identifier
c:\fw\edk2\SampleApp\SampleApp.c(42) : error C2223: left of '->ConIn' must point to struct/union
Building ... c:\fw\edk2\MdeModulePkg\Universal\LoadFileOnFv2\LoadFileOnFv2.inf [IA32]
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\bin\cl.exe"' : return code '0x2'
Stop.
Building ... c:\fw\edk2\MdeModulePkg\Universal\PlatformDriOverrideDxe\PlatformDriOverrideDxe.inf [IA32]
]

build...
: error 7000: Failed to execute command
C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\bin\nmake.exe /nologo tbuild [c:\fw\edk2\Build\NT32IA32\DEBUG_VS2013x86\IA32\SampleApp\SampleApp]
```

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

```
SampleApp.c - Notepad
File Edit Format View Help
#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;
}
```



# Lab 4 : Build and Test SampleApp

At the VS Command Prompt

```
$> Build  
$> RunEmulator.bat
```

Run the application from the shell

```
Shell> SampleApp  
System Table: 0x07E34018  
Press any key to continue:  
Shell>
```

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

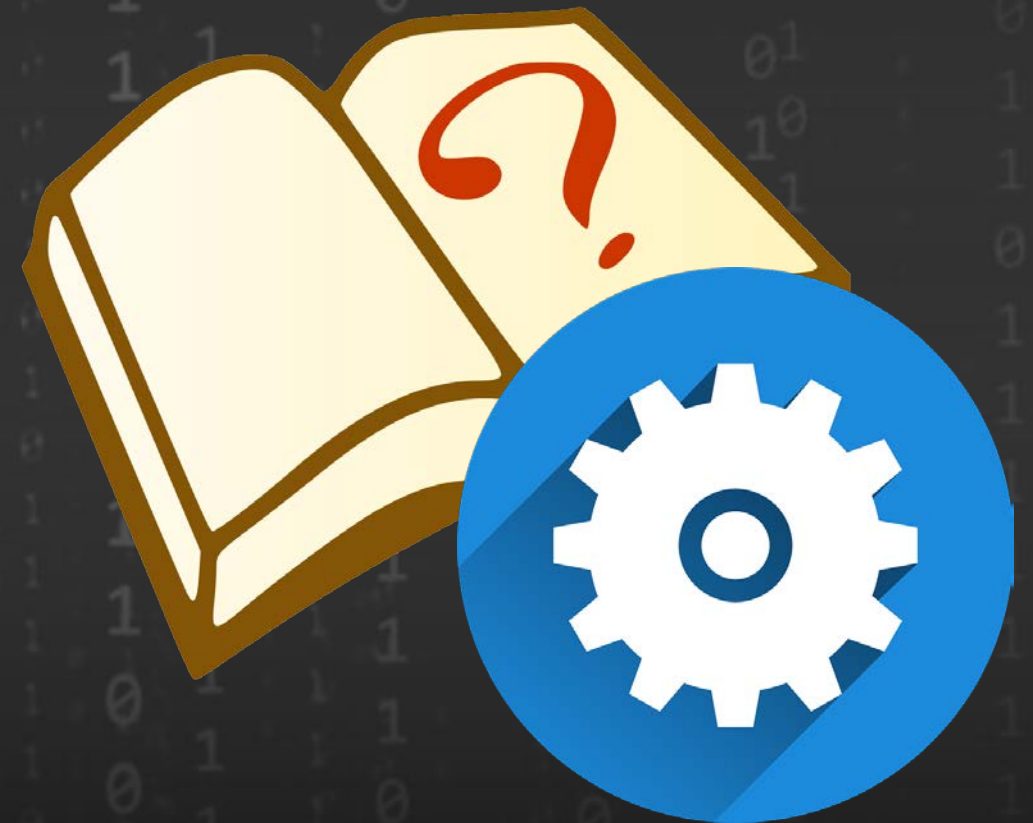
Exit

```
Shell> Reset
```



## 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



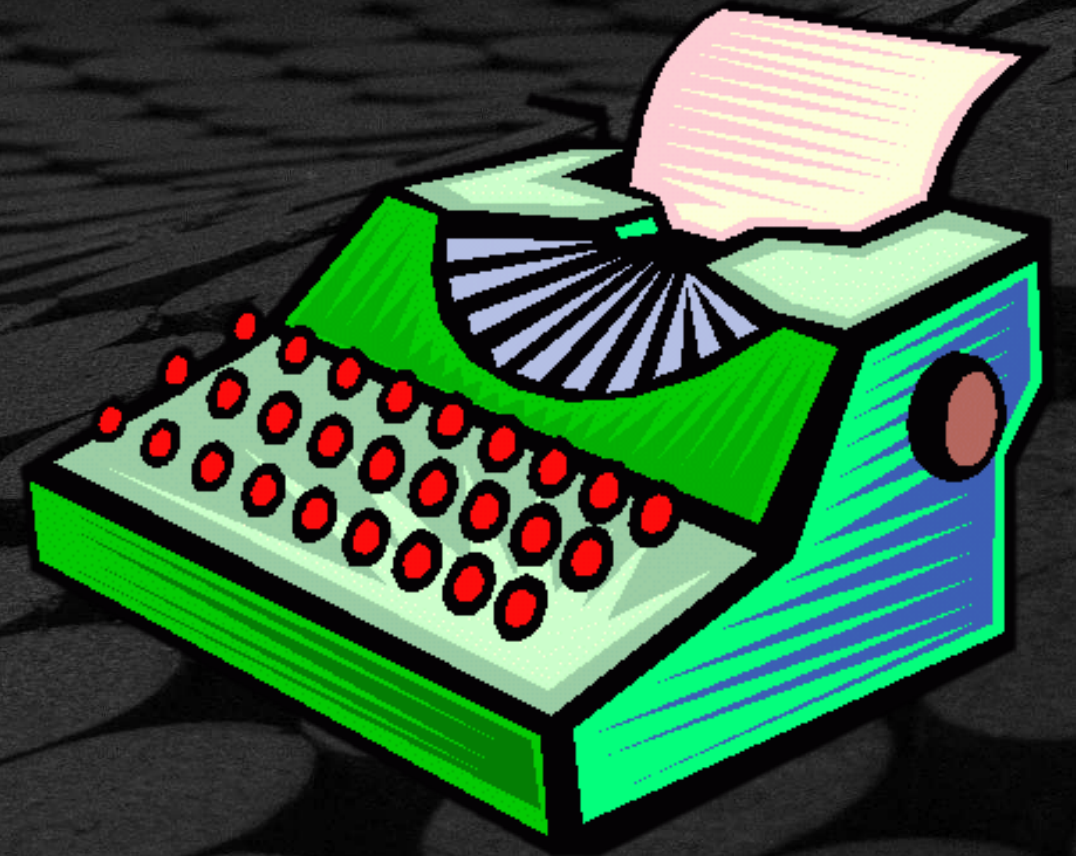


# Lab 5 : Typewriter Function

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>



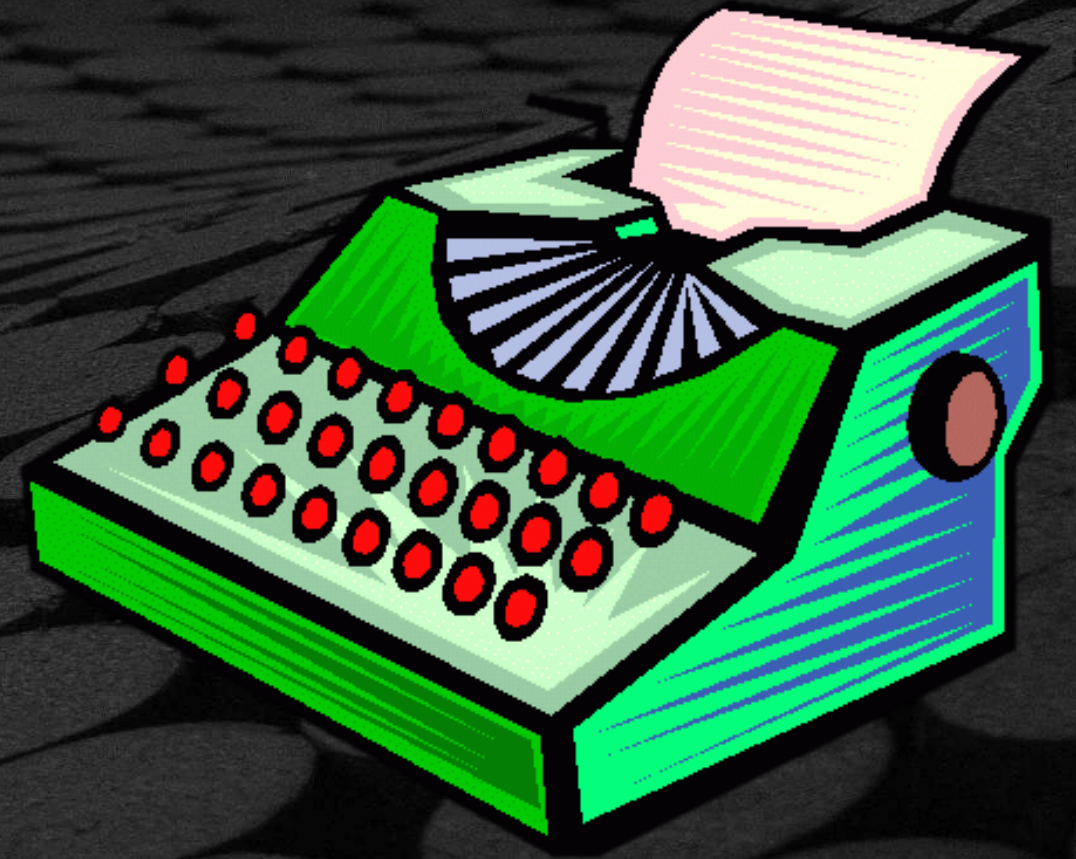


# Lab 5 : Typewriter Function

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





## 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

(hint: Lesson B.5 has the solution)

```

SampleApp.c - Notepad
File Edit Format View Help
#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: 0xp\n",SystemTable);

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

```

```

// 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;
}
```



# Lab 5 :Build and Test SampleApp

At the VS Command Prompt

```
$> Build  
$> RunEmulator.bat
```

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

```
Shell> Reset
```



# 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

```
$> Build  
$> RunEmulator.bat
```

Run the application from the shell

```
Shell> SampleApp test1 test2
```

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



# USING EADK

Using EADK with UEFI Application

**Labs 6-7 are Optional**



## 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

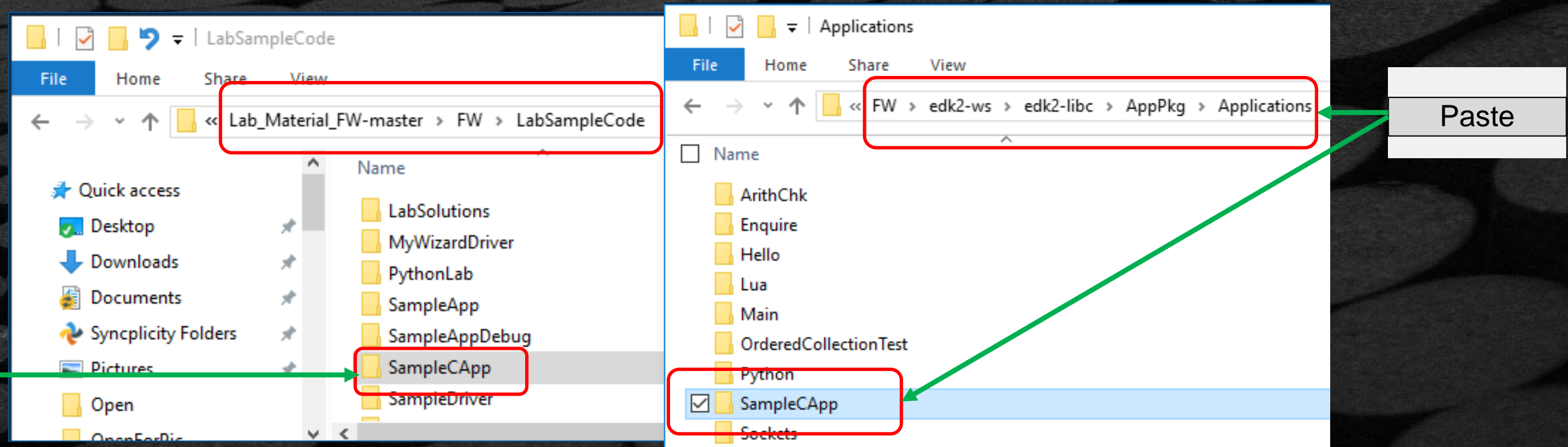
What differences are there using the LibC



# Lab 6: EDK II using EADK

Start with the packages for EADK from edk2-libc

- /edk2-libc - AppPkg - has directory Applications
- /edk2-libc - StdLib - contains the LibC libraries
- Copy and paste directory ../FW/LabSampleCode/SampleCApp to C:/FW/edk2-ws/edk2-libc/AppPkg/Applications/SampleCApp





# Lab 6: EDK II using EADK

Check out AppPkg/Applications/SampleCApp

SampleCApp.c

and

SampleCApp.inf

```
SampleCApp.c - Notepad
File Edit Format View Help

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

```
SampleCApp.inf - Notepad
File Edit Format View Help

[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
```



## Lab 6 : Update AppPkg.dsc

Edit the 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
```



# Lab 6 :Build and Test SampleCApp

Build the AppPkg at the VS Command Prompt

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

Copy the built application to the Emulator runtime directory (note VS Tool)

```
$> copy ..\Build\AppPkg\DEBUG_VS2015x86\X64\SampleCApp.efi  
      ..\Build\EmulatorX64\DEBUG_VS2015x86\X64
```

Run Emulator emulation

```
$> RunEmulator.bat
```

Run the application SampleCApp from the Shell

```
Shell> SampleCApp  
Shell>
```

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



## 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 functionality from Lab 5

SampleCApp.c and

SampleCApp.inf

```

SampleCApp.c - Notepad
File Edit Format View Help

#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;
}

```

```

SampleCApp.inf - Notepad
File Edit Format View Help

[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 functionality from Lab 5

SampleCApp.c and

SampleCApp.inf

```

SampleCApp.c - Notepad
File Edit Format View Help

#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

```

SampleCApp.inf - Notepad
File Edit Format View Help

[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 functionality from Lab 5

SampleCApp.c and

SampleCApp.inf

```

SampleCApp.c - Notepad
File Edit Format View Help

#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

```

SampleCApp.inf - Notepad
File Edit Format View Help

[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 functionality from Lab 5

SampleCApp.c and

SampleCApp.inf

```

SampleCApp.c - Notepad
File Edit Format View Help

#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

```

SampleCApp.inf - Notepad
File Edit Format View Help

[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

```



## SampleCApp.c and SampleCApp.inf

### “C” file

```
#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
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
```



# Lab 7 :Build and Test SampleCApp

## Build the AppPkg at the VS Command Prompt

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

## Copy the built application to the Emulator runtime directory (note VS Tool)

```
$> copy ..\Build\AppPkg\DEBUG_VS2015x86\X64\SampleCApp.efi  
        ..\Build\EmulatorX64\DEBUG_VS2015x86\X64
```

## Run Emulator emulation

```
$> RunEmulator.bat
```

## Run the application SampleCApp from the 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>
```

Copy and paste [LabGuide.md](#)



# SUMMARY

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



# Questions?





# RETURN TO MAIN TRAINING PAGE



Return to Training Table of contents for next presentation [link](#)







# ACKNOWLEDGEMENTS

Redistribution and use in source (original document form) and 'compiled' forms (converted to PDF, epub, HTML and other formats) with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code (original document form) must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.

Redistributions in compiled form (transformed to other DTDs, converted to PDF, epub, HTML and other formats) must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS DOCUMENTATION IS PROVIDED BY TIANOCORE PROJECT "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL TIANOCORE PROJECT BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2021, Intel Corporation. All rights reserved.