

UEFI & EDK II Training

EDK II Modules: Libraries, Drivers & Applications

tianocore.org

LESSON OBJECTIVE

- What is a EDK II Module
- Use EDK II libraries to write UEFI apps/drivers
- How to Define a UEFI application
- Differences between UEFI App / Drivers INF file

EDK II MODULES OVERVIEW

What are EDK II Modules

Smallest separate object compiled in EDK II

Compiles to
.EFI file



Modules: Building blocks of EDK II

Most Used Module Types



Syntax:

```
<ModuleTypes> ::= <ModuleType> [<Space> <ModuleType>]
```


Module Source Contents - minimum file

MODULE_TYPE	Example Source files
UEFI_APPLICATION	Foo.c, Foo.inf
UEFI_DRIVER	FooDriver.c, FooDriver.h, FooDriver.vfr, FooDriver.uni, FooDriver.inf
DXE_DRIVER	Foo.c, Foo.h, Foo.inf
DXE, UEFI or PEIM Library	FooLib.c, FooLib.h, fooLib.inf (w/ LIBRARY_CLASS=)

Complexity - Greater number of source files

.INF file - One file is required per module

.EFI file - Sources compiled to a single .EFI file

EDK II LIBRARY MODULES

Syntax:

```
[LibraryClasses.common]  
  <LibraryClassName> | <LibraryInstancePathToInf/Name.inf>
```

```
DebugLib|MdePkg/Library/BaseDebugLibNull/BaseDebugLibNull.inf
```

Name

Implementation³

Consistent set of interfaces

Does not describe implementation of the interfaces

“NULL” Library Class

Constructors

Special Cases

NOT “. . . LibNull” instance

Syntax

```
Pkg/MyModule/MyModule.inf {  
  <LibraryClasses>  
    NULL|Pkg/Library/LibName/LibName.inf  
    NULL|Pkg/Library/LibName2/LibName2.inf  
}
```

Open Source Example

DxeCrc32GuidedSectionExtractLib
ShellPkg as used with Profiles

UEFI Shell example:

```
ShellPkg/Application/Shell/Shell.inf {  
  <LibraryClasses>  
    NULL|ShellPkg/Library/UefiShellDriver1CommandsLib/UefiShellDriver1CommandsLib.inf  
    NULL|ShellPkg/Library/UefiShellNetwork1CommandsLib/UefiShellNetwork1CommandsLib.inf  
    . . .  
}
```

Locating Library Classes

Library based upon

1. Industry specs (UEFI, etc.)

MdePkg/MdeModulePkg

2. Features

NetworkPkg/SecurityPkg

Use the package help files (.CHM) to find a library or function

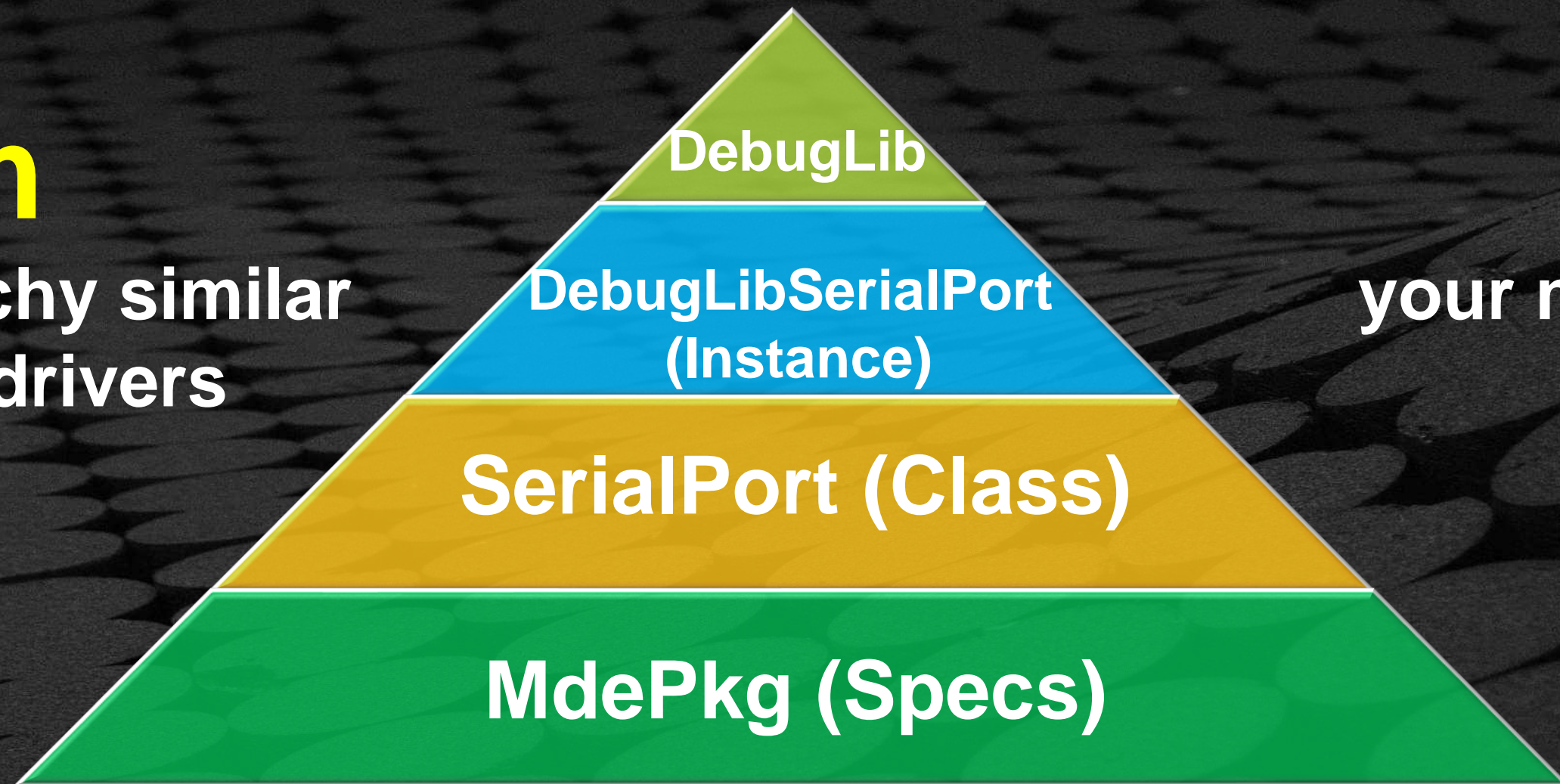
Example: MdePkg.chm

Search WorkSpace (.INF) "LIBRARY_CLASS"

Library Instance Hierarchy

Form

a hierarchy similar
to UEFI drivers



Link

your module to
another

Build error : Instance of Library class [*Foo...Lib*] is not found
Consumed by module [*My Module.inf*]

Commonly Used Base Library Classes

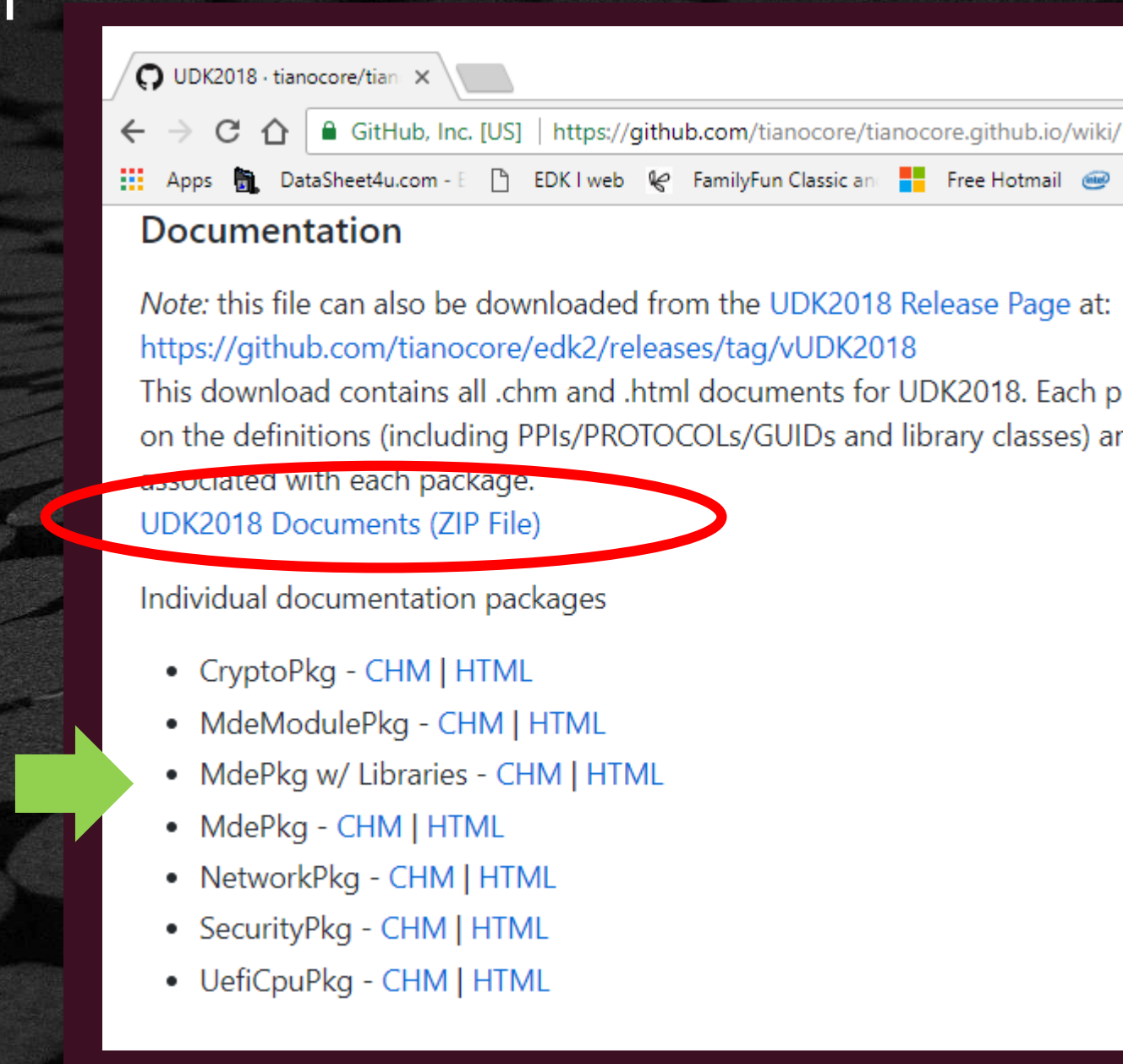
BaseLib DebugLib UefiDriverEntryPoint
UefiLib UefiBootServicesTableLib
UefiApplicationEntryPoint DxeCoreEntryPoint
CpuLib UefiUsbLib PciLib DevicePathLib IoLib
MemoryAllocationLib PrintLib PeimEntryPoint
PeiCoreEntryPoint UefiScsiLib BaseMemoryLib
UefiRuntimeLib SmmMemLib
DxeServicesLib SynchronizationLib PciExpressLib
DxePcdLib UefiRuntimeServicesTableLib
PciSegmentLibLib PeiServicesLib
PeiPcdLib DxeHobLib UefiFileHandleLib

MdePkg Library .CHM file Location

tianocore.org UDK2018 documentation on

 [Latest UDK Release](#)

 [UDK2018](#)



UDK2018 · tianocore/tianocore.github.io

← → ↻ 🏠 🔒 GitHub, Inc. [US] | <https://github.com/tianocore/tianocore.github.io/wiki/>

Apps DataSheet4u.com - E EDK I web FamilyFun Classic an Free Hotmail intel

Documentation

Note: this file can also be downloaded from the [UDK2018 Release Page](#) at: <https://github.com/tianocore/edk2/releases/tag/vUDK2018>

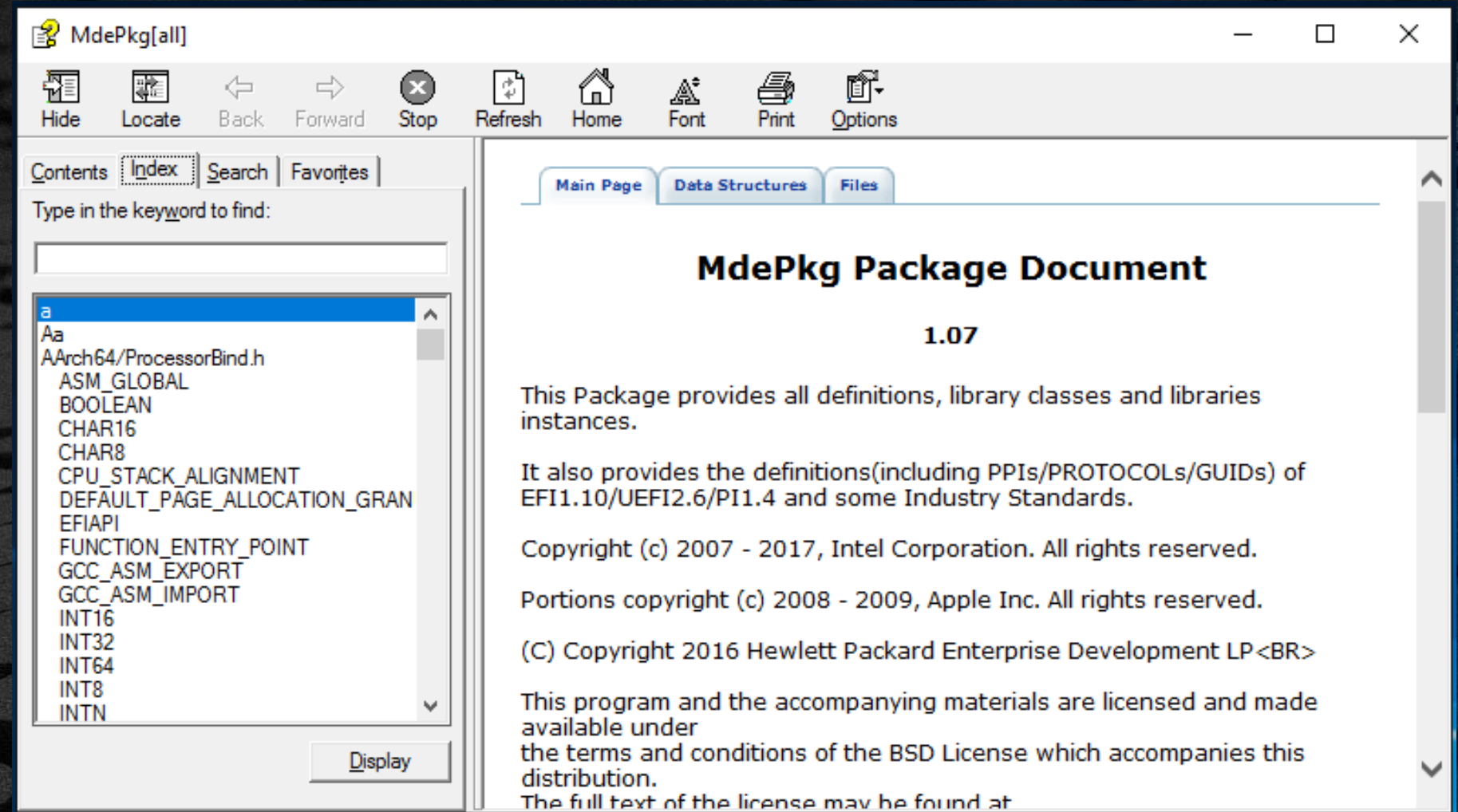
This download contains all .chm and .html documents for UDK2018. Each package contains the definitions (including PPIs/PROTOCOLs/GUIDs and library classes) and the associated with each package.

[UDK2018 Documents \(ZIP File\)](#)

Individual documentation packages

- [CryptoPkg - CHM | HTML](#)
- [MdeModulePkg - CHM | HTML](#)
- [MdePkg w/ Libraries - CHM | HTML](#)
- [MdePkg - CHM | HTML](#)
- [NetworkPkg - CHM | HTML](#)
- [SecurityPkg - CHM | HTML](#)
- [UefiCpuPkg - CHM | HTML](#)

Library Navigation Demonstration

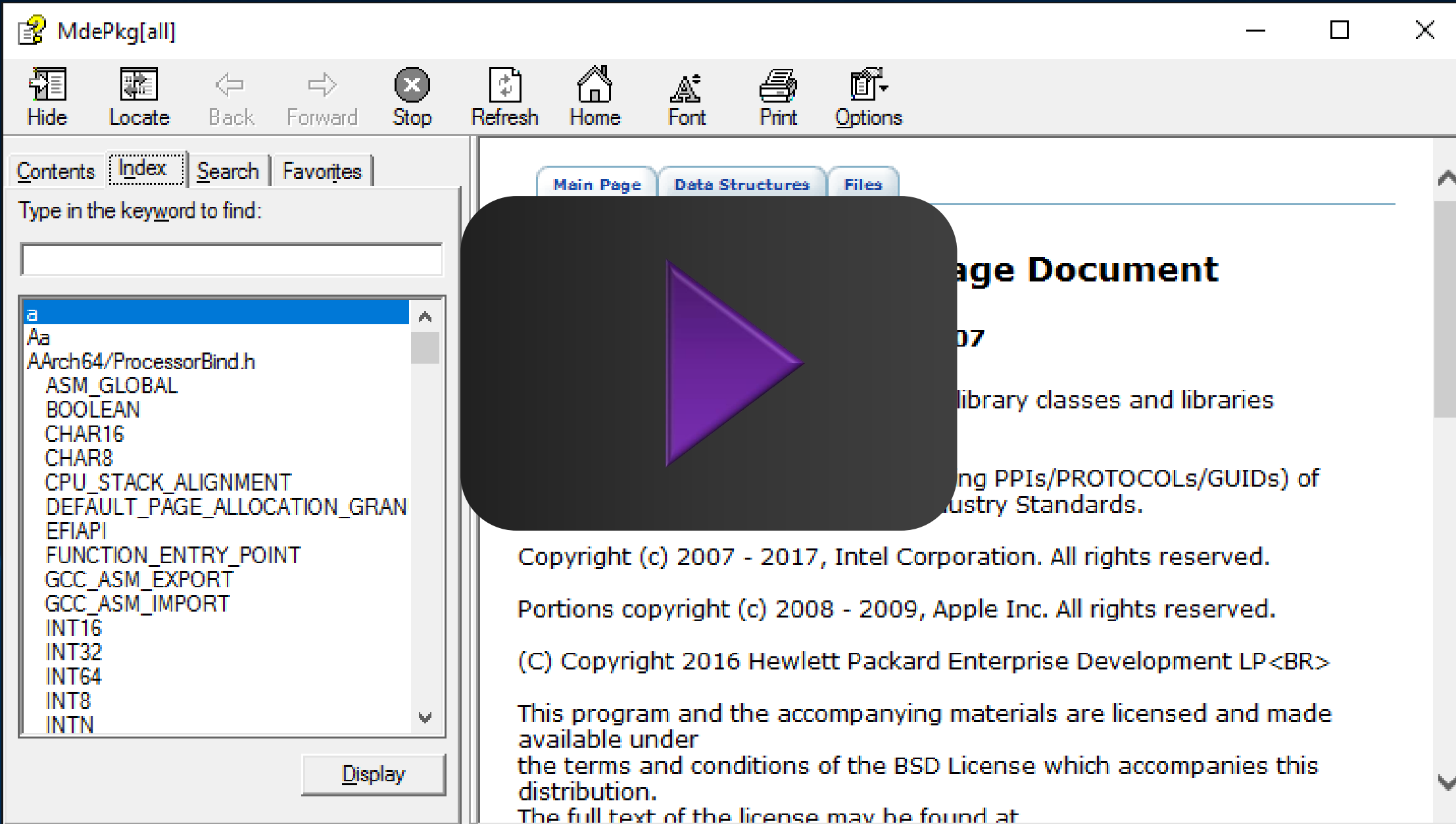


Open file: /FW/Documentation/"MdePkg Document With LibrariesMdePkg.chm"

NOTE: Install a CHM Viewer for Ubuntu

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


Library Navigation Demonstration



The screenshot shows the MdePkg[all] library navigation interface. The window title is "MdePkg[all]". The top toolbar includes icons for Hide, Locate, Back, Forward, Stop, Refresh, Home, Font, Print, and Options. The left sidebar has tabs for Contents, Index, Search, and Favorites. Below the tabs is a search bar labeled "Type in the keyword to find:". Below the search bar is a list of symbols, including "a", "Aa", "AArch64/ProcessorBind.h", "ASM_GLOBAL", "BOOLEAN", "CHAR16", "CHAR8", "CPU_STACK_ALIGNMENT", "DEFAULT_PAGE_ALLOCATION_GRAN", "EFIAPI", "FUNCTION_ENTRY_POINT", "GCC_ASM_EXPORT", "GCC_ASM_IMPORT", "INT16", "INT32", "INT64", "INT8", and "INTN". A "Display" button is at the bottom of the list. The main area shows the "Main Page" document with tabs for Main Page, Data Structures, and Files. The document content includes copyright information: "Copyright (c) 2007 - 2017, Intel Corporation. All rights reserved.", "Portions copyright (c) 2008 - 2009, Apple Inc. All rights reserved.", "(C) Copyright 2016 Hewlett Packard Enterprise Development LP
", and "This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at". A large play button is overlaid on the center of the screenshot.

<https://youtu.be/s8Zw1w1iQS4>

EDK II UEFI APPLICATION

Defining a UEFI Application

Characteristics of a UEFI Loadable Image

- ✱ Loaded by UEFI loader, just like drivers
- ✱ Does not register protocols
- ✱ Consumes protocols
- ✱ Typically exits when completed (user driven)
- ✱ Same set of interfaces as drivers available

Defining a UEFI Application

UEFI Loadable Image Usages

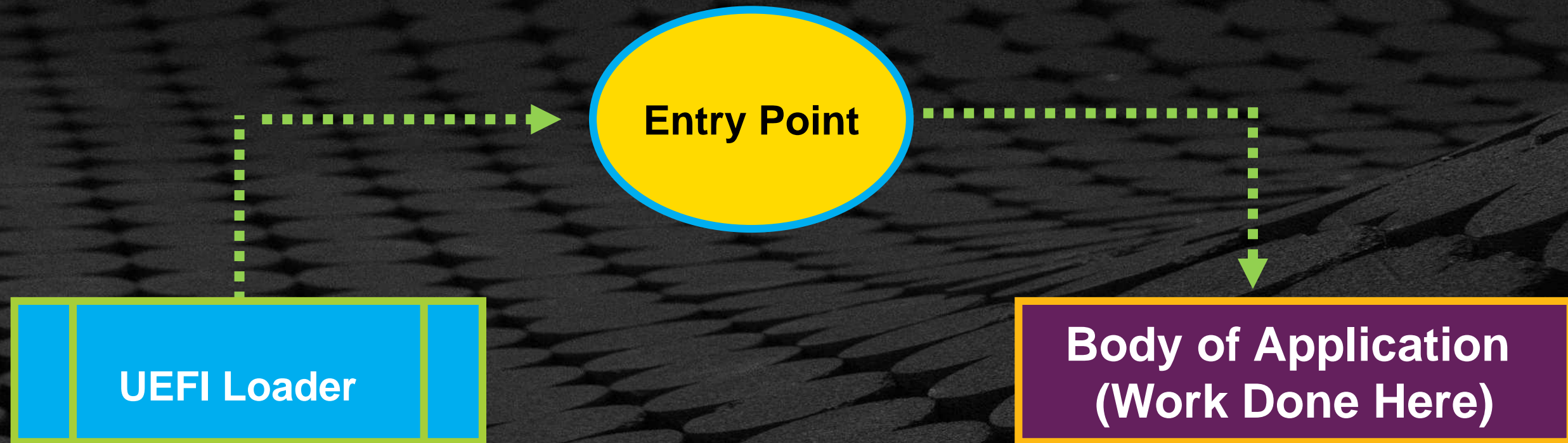
- ✱ Platform Diagnostics
- ✱ Factory Diagnostics
- ✱ Utilities
- ✱ Driver Prototyping
- ✱ “Platform” Applications
- ✱ Portable Across Platforms (IA32, X64, ARM, Itanium, etc.)

Executing Applications

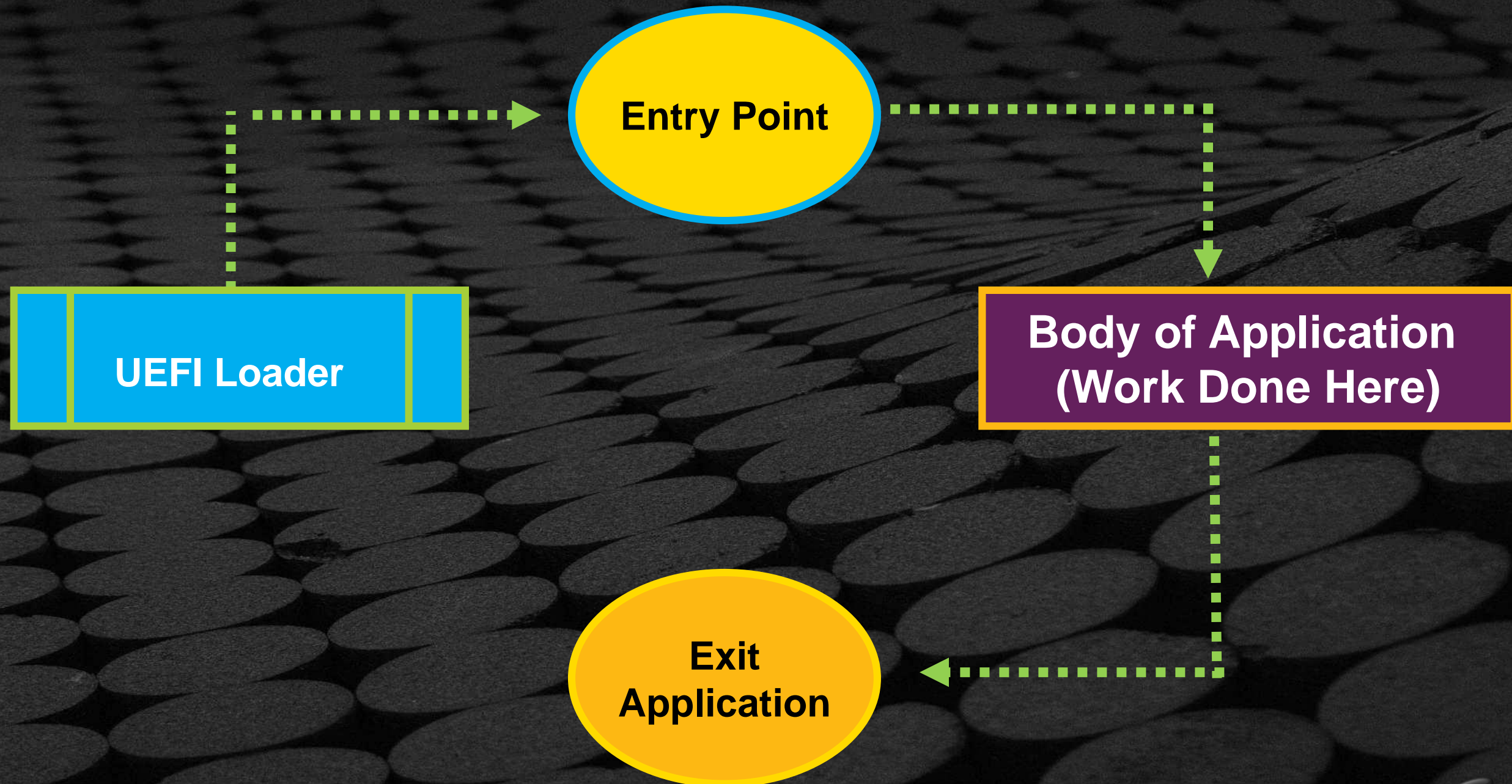


UEFI Loader

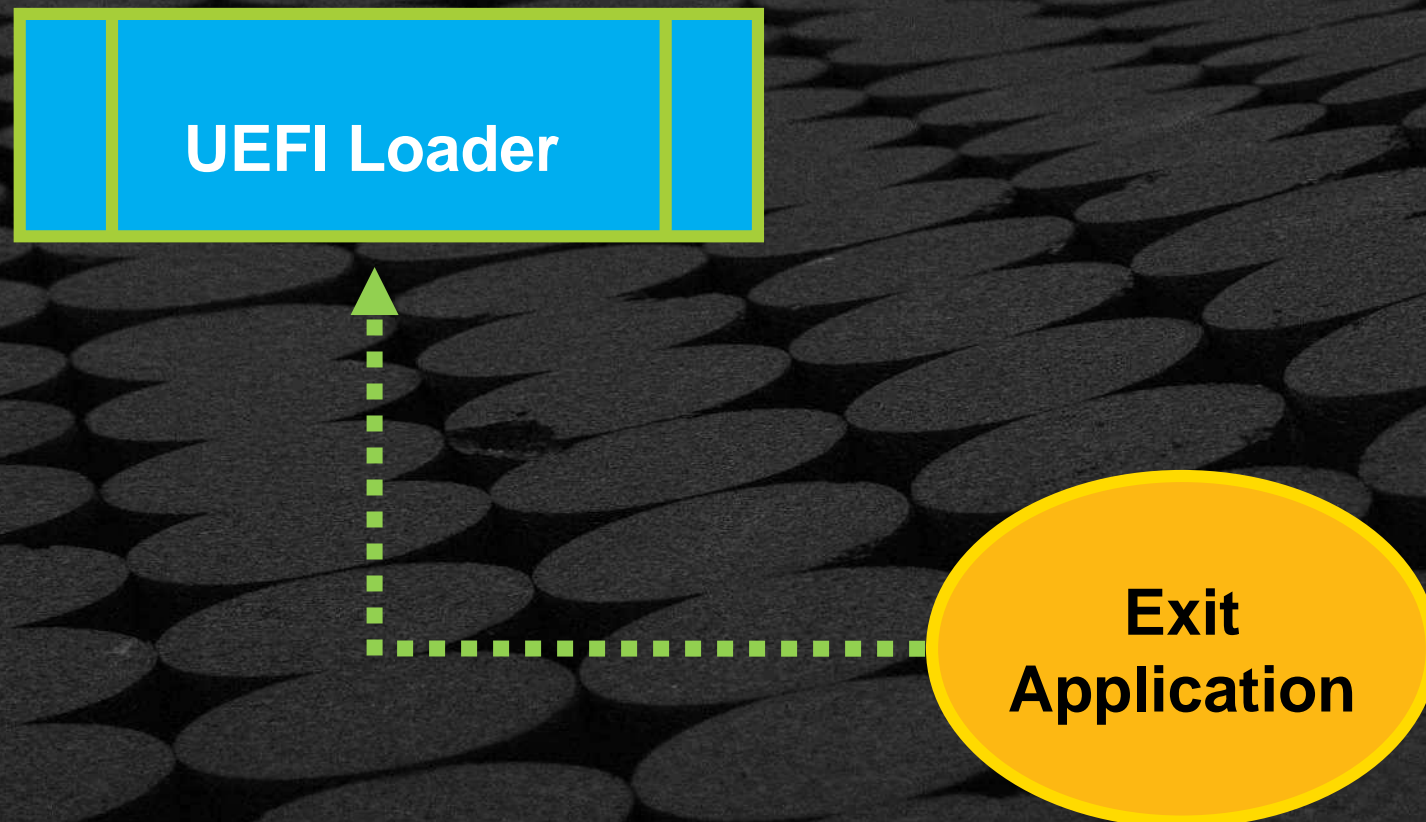
Executing Applications



Executing Applications



Executing Applications



Executing Applications



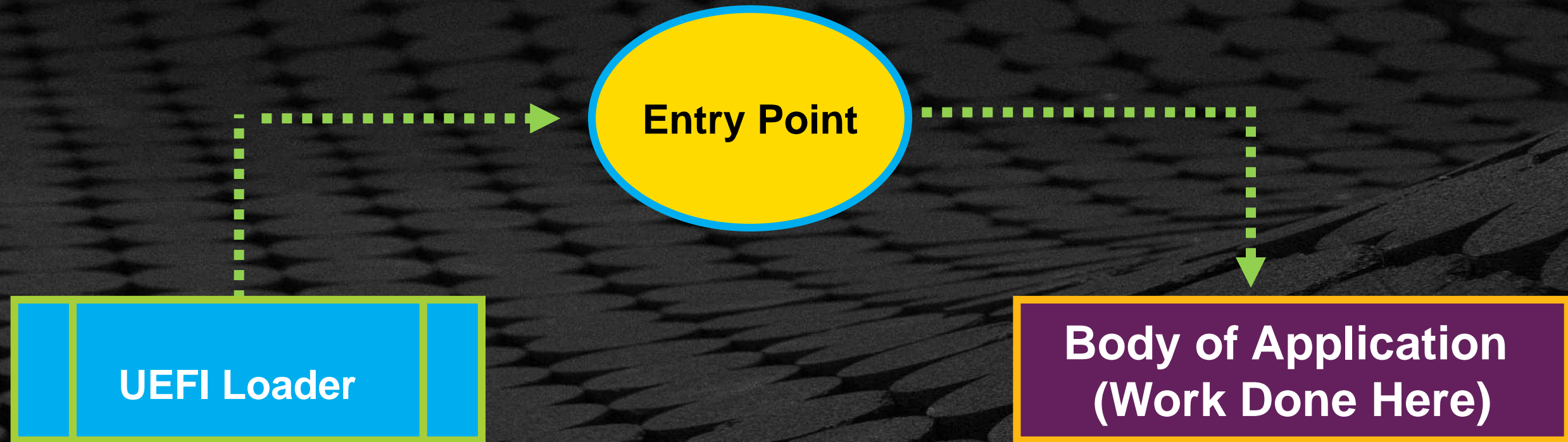
UEFI Loader

Executing Applications



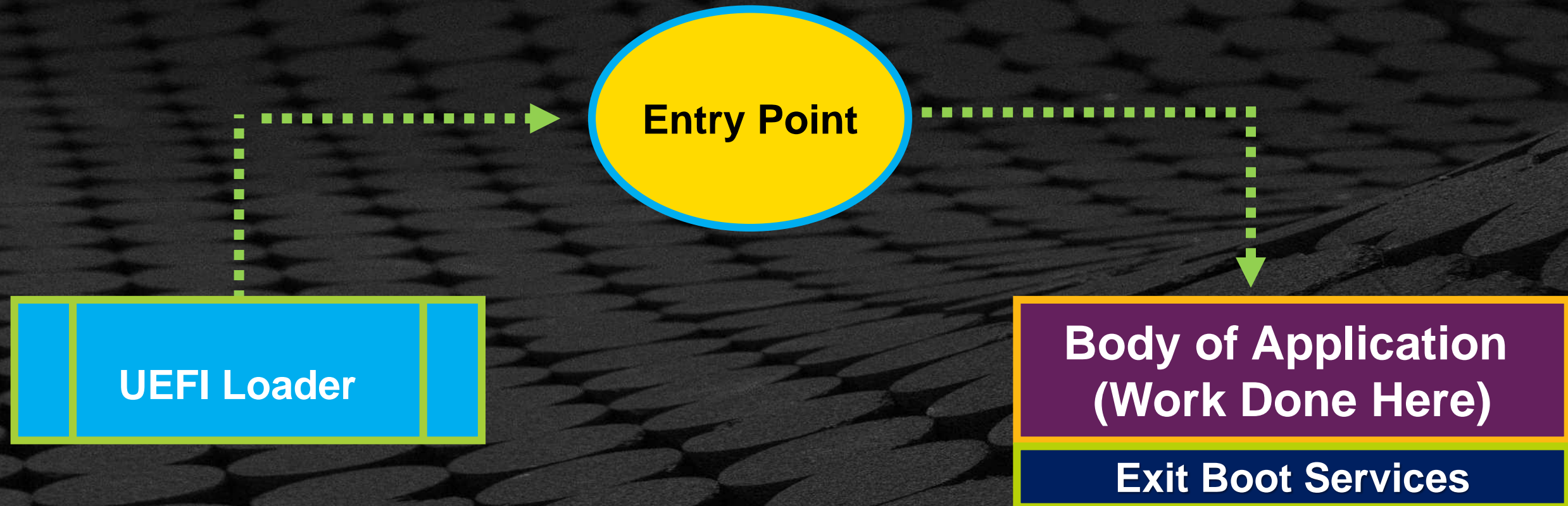
OS Loader

Executing Applications



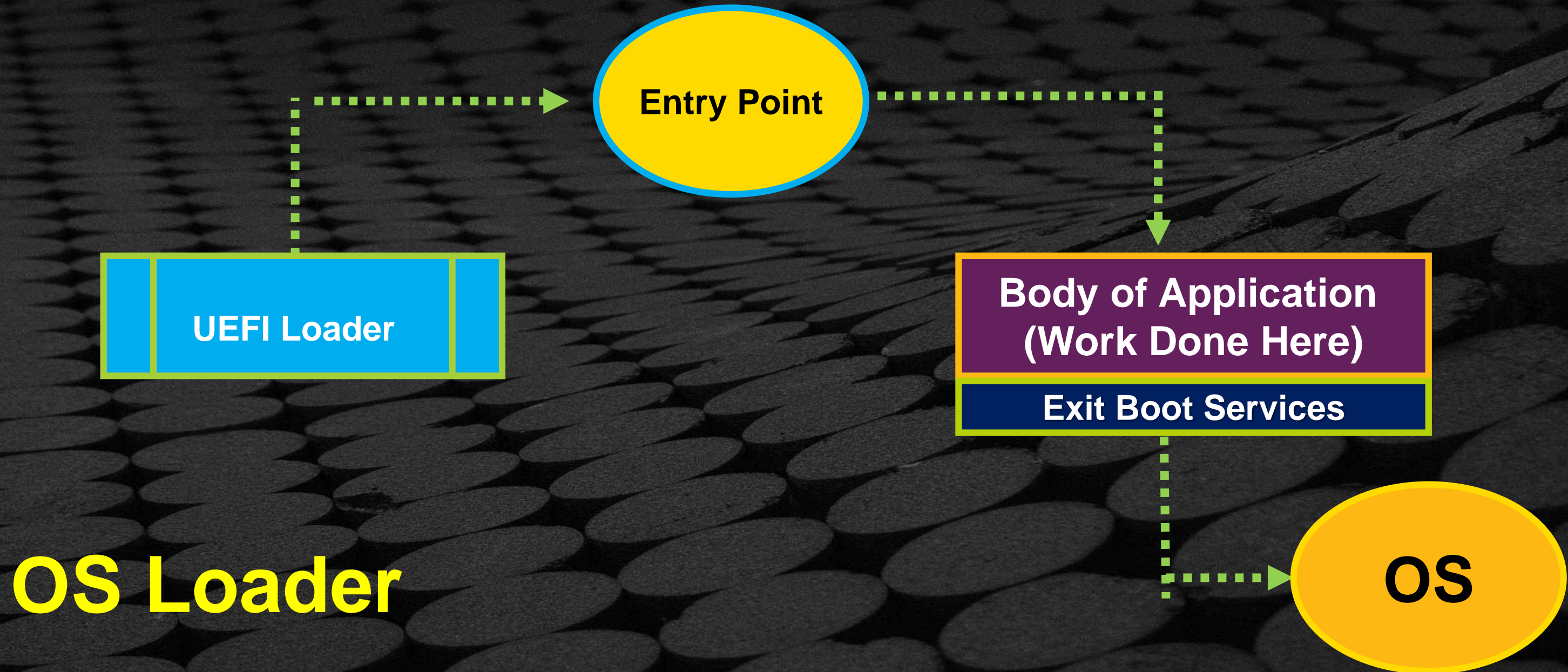
OS Loader

Executing Applications

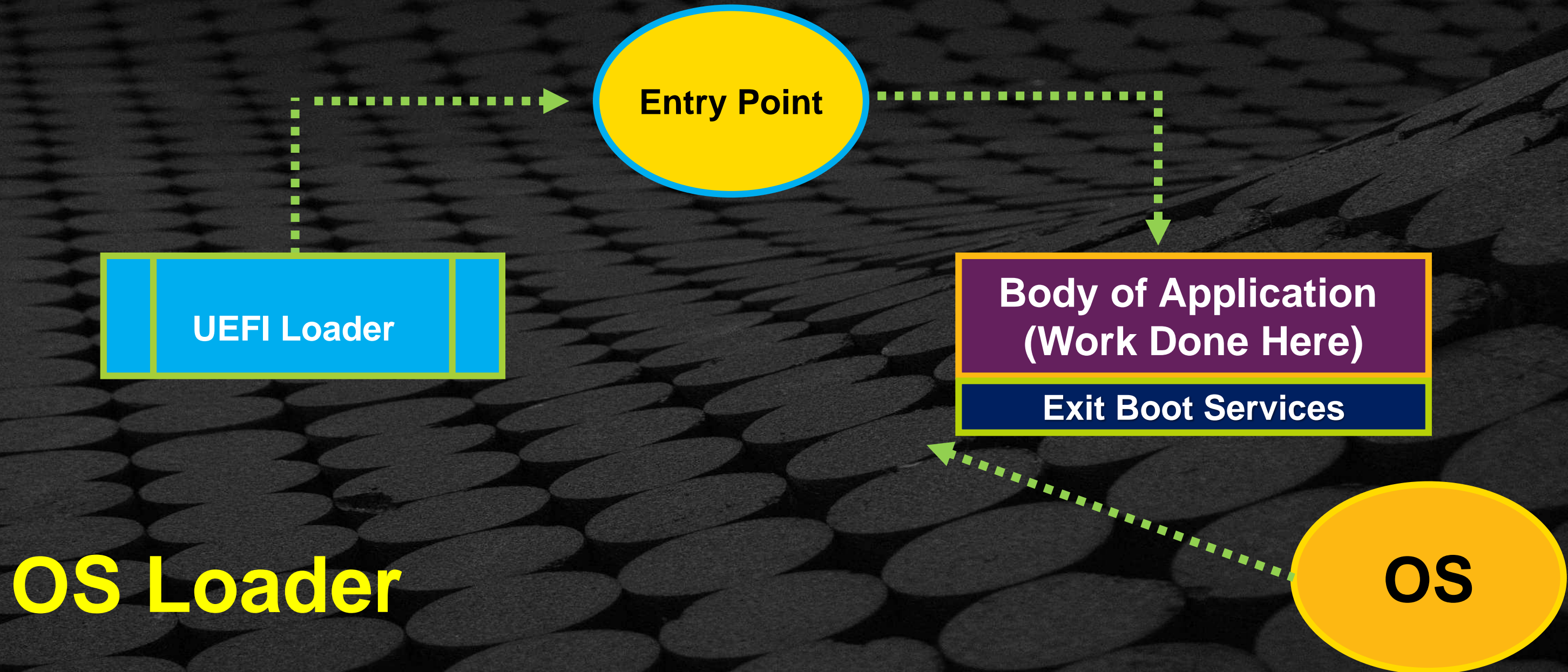


OS Loader

Executing Applications



Executing Applications



Executing Applications



OS

OS Loader

Driver Vs. Application

	Driver	Application
Loaded by:	UEFI Loader	UEFI Loader
Interfaces available:	ALL	ALL
Consume protocols?	YES	YES
Produce protocols?		
Typically driven by?		
Typical use		

Driver Vs. Application

	Driver	Application
Loaded by:	UEFI Loader	UEFI Loader
Interfaces available:	ALL	ALL
Consume protocols?	YES	YES
Produce protocols?	YES	NO
Typically driven by?	System	User
Typical use	Support Hardware	Any

EDK II UEFI APPLICATIONS

How to Write a EDK II UEFI Application


Application Files Placement

- Application source files can be located anywhere in the EDK II workspace including PACKAGES_PATH
- All code and include files go under a single directory containing the driver INF
- EDK II Sample Applications can be found here:

`edk2/MdeModulePkg/Application`

- Typically, modules reside within a package:

```
MyWorkSpace/  
  edk2/  
    MyPkg/  
      Application/  
        MyApp/
```



`MyApp.c`
`MyApp.inf`

Module .INF File

Syntax

INF text file example

```
INFfile ::= [<Header>
            <Defines>
            [<BuildOptions>]
            [<Sources>]
            [<Binaries>]
            [<Guids>]
            [<Protocols>]
            [<Ppis>]
            [<Packages>]
            [<LibraryClasses>]
            [<Pcds>]
            [<UserExtensions>]
            [<Depex>]
```


Application INF Files [DEFINES]

Field	Description
INF_VERSION	1.25* - Version of the INF spec.
BASE_NAME	What's the name of the application
FILE_GUID	Create a GUID for your module
MODULE_UNI_FILE	Meta-data - localization for Description & Abstract
VERSION_STRING	Version number
ENTRY_POINT	Name of the function to call
MODULE_TYPE	UEFI_APPLICATION

* EDK II Specifications: <https://github.com/tianocore/tianocore.github.io/wiki/EDK-II-Specifications>

Sample INF file

```
[Defines]
  INF_VERSION           = 0x00010005
  BASE_NAME             = MyApplication
  MODULE_UNI_FILE       = MyFile.uni
  FILE_GUID             = 10C75C00-30 . . .
  MODULE_TYPE           = UEFI_APPLICATION
  VERSION_STRING        = 1.0
  ENTRY_POINT           = UefiMain
[Sources]
  MyFile.c

[Packages]
  MdePkg/MdePkg.dec

[LibraryClasses]
  UefiApplicationEntryPoint

[Guids]

[Ppis]

[Protocols]
```


Sample INF file

```
[Defines]
  INF_VERSION          = 0x00010005
  BASE_NAME            = MyApplication
  MODULE_UNI_FILE      = MyFile.uni
  FILE_GUID            = 10C75C00-30 . . .
  MODULE_TYPE          = UEFI_APPLICATION
  VERSION_STRING       = 1.0
  ENTRY_POINT          = UefiMain
```

```
[Sources]
  MyFile.c
```

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

```
[LibraryClasses]
  UefiApplicationEntryPoint
```

```
[Guids]
```

```
[Ppis]
```


BUILDING AN APPLICATION

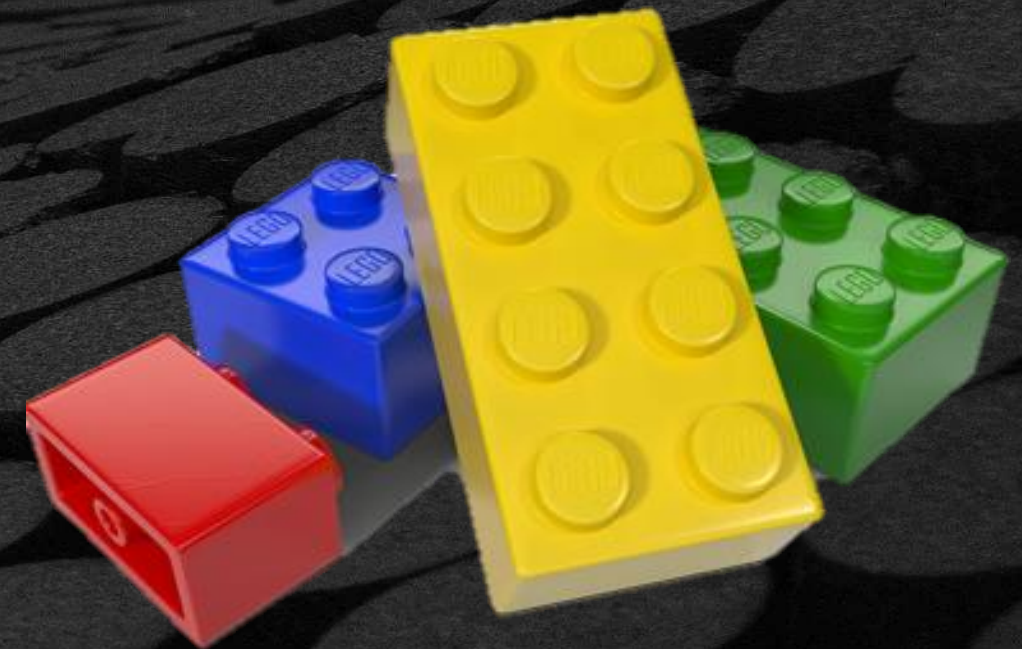
Platform .DSC references .INF

Runs:

“Build” for the entire platform

OR

“Build” in the application’s directory



Sample Application 'C' file

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

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


Sample Application 'C' file

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

```
EFI_STATUS
```

```
EFI_API
```

```
UefiMain (
```

```
    IN EFI_HANDLE
```

```
    IN EFI_SYSTEM_TABLE
```

```
)
```

```
{
```

```
    return EFI_SUCCESS;
```

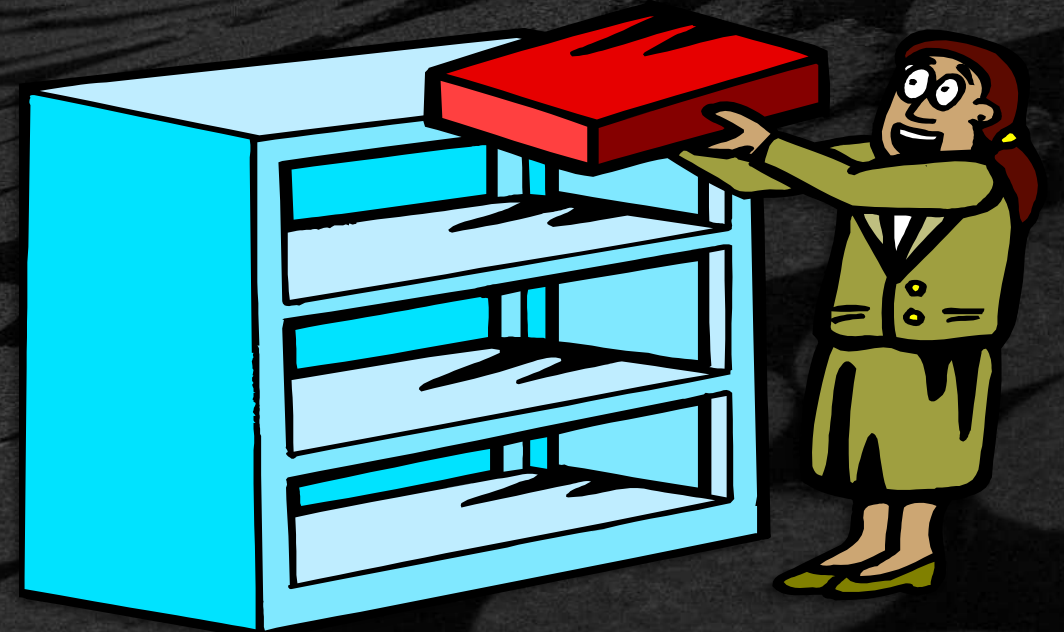
```
}
```

```
    ImageHandle,  
    *SystemTable
```


UEFI Application Vs. EADK Application

EDK II Application Development Kit
includes the Standard C Libraries in
UEFI Shell Applications

Off the shelf “C” application
Converted to UEFI application



Sample INF file using EDK II EADK

```
[Defines]
  INF_VERSION           = 0x00010005
  BASE_NAME             = MyApplication
  FILE_GUID             = 10C75C00-30 . . .
  MODULE_TYPE           = UEFI_APPLICATION
  VERSION_STRING        = 1.0
  ENTRY_POINT           = ShellCEntryLib
[Sources]
  MyFile.c

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

[LibraryClasses]
  LibC
  LibStdio
```


Sample INF file using EDK II EADK

```
[Defines]
  INF_VERSION           = 0x00010005
  BASE_NAME             = MyApplication
  FILE_GUID             = 10C75C00-30 . . .
  MODULE_TYPE           = UEFI_APPLICATION
  VERSION_STRING        = 1.0
  ENTRY_POINT           = ShellCEntryLib
```

```
[Sources]
  MyFile.c
```

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

```
[LibraryClasses]
  LibC
  LibStdio
```


Sample Application 'C' file Using EDK II EADK

This sample looks a lot like actual "C" source.

```
#include <stdio.h>
```

```
int  
Main (  
    IN int Argc,  
    IN char **Argv  
)  
{  
    return 0;  
}
```

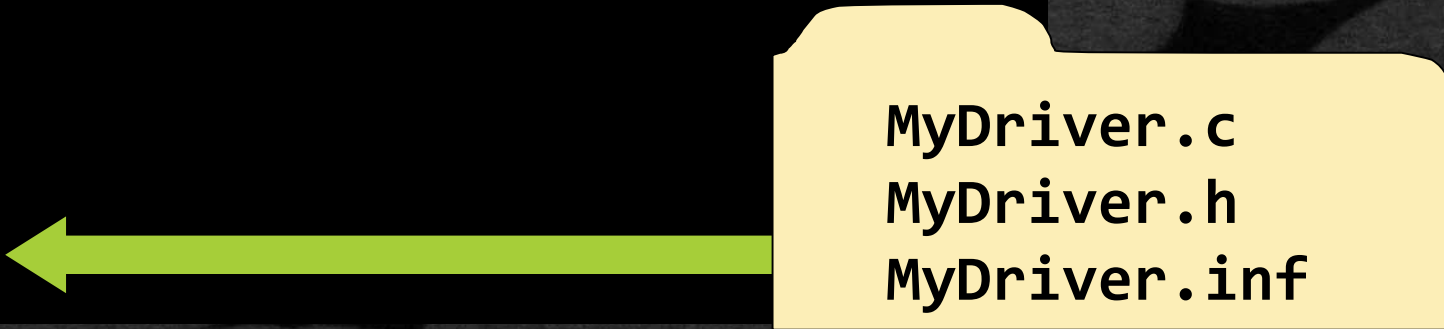

Driver Files Placement

- ★ Driver source code can go anywhere in the EDK II workspace
- ★ All code and include files go under a single directory containing an INF
- ★ Good example of UEFI Drivers can be found here:

`edk2/MdeModulePkg/Bus/ScsiDiskDxe`

- ★ Typically, Driver modules reside within a package:

```
MyWorkSpace/  
  edk2/  
    MyPkg/  
      Include/  
        MyDriver/
```



```
MyDriver.c  
MyDriver.h  
MyDriver.inf
```


Driver INF Files: [DEFINES]

Field	Description
INF_VERSION	1.25* - Version of the INF spec.
BASE_NAME	What's the name of the driver
FILE_GUID	Create a GUID for your module
MODULE_UNI_FILE	Meta-data - localization for Description & Abstract
VERSION_STRING	Version number
ENTRY_POINT	Name of the function to call
MODULE_TYPE	UEFI_DRIVER, DXE_DRIVER, PEIM, or others

Changes for a UEFI Driver Module

Applications can be converted to a driver

But ...It remains in memory after it runs

UEFI Driver Module requirements:

- Driver Binding Protocol
- Component Name2 Protocol (recommended

DXE/PEIM/other Driver requirements



Sample Driver INF file

```
[Defines]
  INF_VERSION          = 0x00010005
  BASE_NAME            = MyDriver
  FILE_GUID            = 10C75C00-30 . . .
  MODULE_TYPE          = UEFI_DRIVER
  VERSION_STRING       = 1.0
  ENTRY_POINT          = UefiMain
```

```
[Sources]
  MyDriverFile.c
```

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

```
[LibraryClasses]
  UefiDriverEntryPoint
  . . .
```

```
[Guids]
  . . .
[Protocols]
  . . .
```


INF Usage Fields – DIST files

Optional UEFI Spec – Package Distribution

Usage Fields used by Build tools for creating the .Dist files for binary modules

[GUID]

[PCD]

[PROTOCOL]

[PPIS]

1 Usage Block – “##” After the entry

n Usage Blocks – “##” Precede the entry

Usage Key Word

UNDEFINED

CONSUMES

SOMETIMES_CONSUMES

PRODUCES

SOMETIMES_PRODUCES

TO_START

BY_START

NOTIFY

} UEFI Protocol



INF File Usage Block examples

[Guids]

```
## SOMETIMES_PRODUCES ## Variable:L"ConInDev"  
## SOMETIMES_CONSUMES ## Variable:L"ConInDev"  
## SOMETIMES_PRODUCES ## Variable:L"ConOutDev"  
## SOMETIMES_CONSUMES ## Variable:L"ConOutDev"  
## SOMETIMES_PRODUCES ## Variable:L"ErrOutDev"  
## SOMETIMES_CONSUMES ## Variable:L"ErrOutDev"  
gEfiGlobalVariableGuid
```

```
gEfiVTUTF8Guid ## SOMETIMES_CONSUMES ## GUID # used with a Vendor-Defined  
gEfiVT100Guid ## SOMETIMES_CONSUMES ## GUID # used with a Vendor-Defined  
gEfiVT100PlusGuid ## SOMETIMES_CONSUMES ## GUID # used with a Vendor-Defined  
gEfiPcAnsiGuid ## SOMETIMES_CONSUMES ## GUID # used with a Vendor-Defined  
gEfiTtyTermGuid ## SOMETIMES_CONSUMES ## GUID # used with a Vendor-Defined  
gEdkiiStatusCodeDataTypeVariableGuid ## SOMETIMES_CONSUMES ## GUID
```

Example: [TerminalDxe.inf](#)

INF File Usage Block examples

[Protocols]

gEfiSerialIoProtocolGuid ## TO_START

BY_START

TO_START

gEfiDevicePathProtocolGuid

gEfiSimpleTextInProtocolGuid ## BY_START

gEfiSimpleTextInputExProtocolGuid ## BY_START

gEfiSimpleTextOutProtocolGuid ## BY_START

[Pcd]

gEfiMdePkgTokenSpaceGuid.PcdDefaultTerminalType ## SOMETIMES_CONSUMES

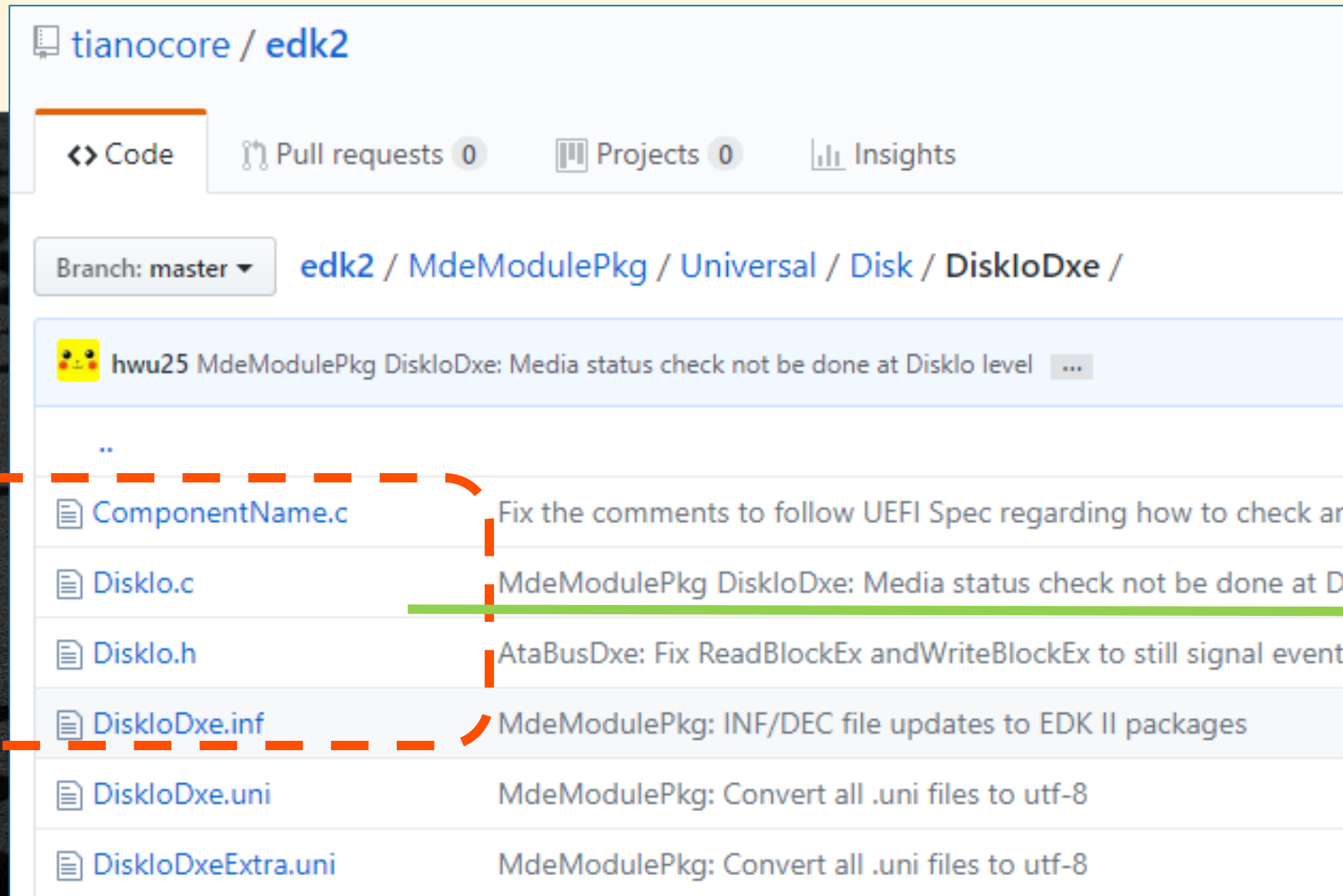
gEfiMdeModulePkgTokenSpaceGuid.PcdErrorCodeSetVariable ## CONSUMES

Example: [TerminalDxe.inf](#)

UEFI Driver Example – Disk I/O



<https://github.com/tianocore/edk2/MdeModulePkg/Universal/Disk/DiskIoDxe>



tianocore / edk2

Code Pull requests 0 Projects 0 Insights

Branch: master edk2 / MdeModulePkg / Universal / Disk / DiskIoDxe /

hww25 MdeModulePkg DiskIoDxe: Media status check not be done at DiskIo level ...

..

ComponentName.c	Fix the comments to follow UEFI Spec regarding how to check an
DiskIo.c	MdeModulePkg DiskIoDxe: Media status check not be done at Di
DiskIo.h	AtaBusDxe: Fix ReadBlockEx andWriteBlockEx to still signal event
DiskIoDxe.inf	MdeModulePkg: INF/DEC file updates to EDK II packages
DiskIoDxe.uni	MdeModulePkg: Convert all .uni files to utf-8
DiskIoDxeExtra.uni	MdeModulePkg: Convert all .uni files to utf-8

Driver Binding
Supported
Start
Stop

UEFI Driver Example – Disk I/O

 <https://github.com/tianocore/edk2/.../Disk/DiskIoDxe>

Entry Point

“C” File

```
EFI_STATUS
EFI_API
InitializeDiskIo (
    IN EFI_HANDLE      ImageHandle,
    IN EFI_SYSTEM_TABLE *SystemTable
)
{
    // ...

    Status = EfiLibInstallDriverBindingComponentName2 (
        ImageHandle,
        SystemTable,
        &gDiskIoDriverBinding,
        ImageHandle,
        &gDiskIoComponentName,
        &gDiskIoComponentName2
    );
    ASSERT_EFI_ERROR (Status);
    return Status;
}
```

INF File

```
[Defines]

ENTRY_POINT = InitializeDiskIo
```


UEFI Driver Example – Disk I/O

 <https://github.com/tianocore/edk2/.../Disk/DiskIoDxe>

Supported

“C” File

```
EFI_STATUS
EFI_API
DiskIoDriverBindingSupported (
    IN EFI_DRIVER_BINDING_PROTOCOL *This,
    IN EFI_HANDLE ControllerHandle,
    IN EFI_DEVICE_PATH_PROTOCOL *RemainingDevicePath
    OPTIONAL
)
{
    Status = gBS->OpenProtocol (
        ControllerHandle,
        &gEfiBlockIoProtocolGuid,
        (VOID **) &BlockIo,
        This->DriverBindingHandle,
        ControllerHandle,
        EFI_OPEN_PROTOCOL_BY_DRIVER
    );
}
```

INF File

[Protocols]

```
gEfiBlockIoProtocolGuid  ## TO_START
```


UEFI Driver Example – Disk I/O

 <https://github.com/tianocore/edk2/.../Disk/DiskIoDxe>

Start

“C” File

```
EFI_STATUS
EFI_API
DiskIoDriverBindingStart (
    IN EFI_DRIVER_BINDING_PROTOCOL *This,
    IN EFI_HANDLE
    IN EFI_DEVICE_PATH_PROTOCOL *RemainingDevicePath
    OPTIONAL
)
{
    if (Instance->BlockIo2 != NULL) {
        Status = gBS->InstallMultipleProtocolInterfaces (
            &ControllerHandle,
            &gEfiDiskIoProtocolGuid, &Instance->DiskIo,
            &gEfiDiskIo2ProtocolGuid, &Instance->DiskIo2,
            NULL
        );
    }
}
```

INF File

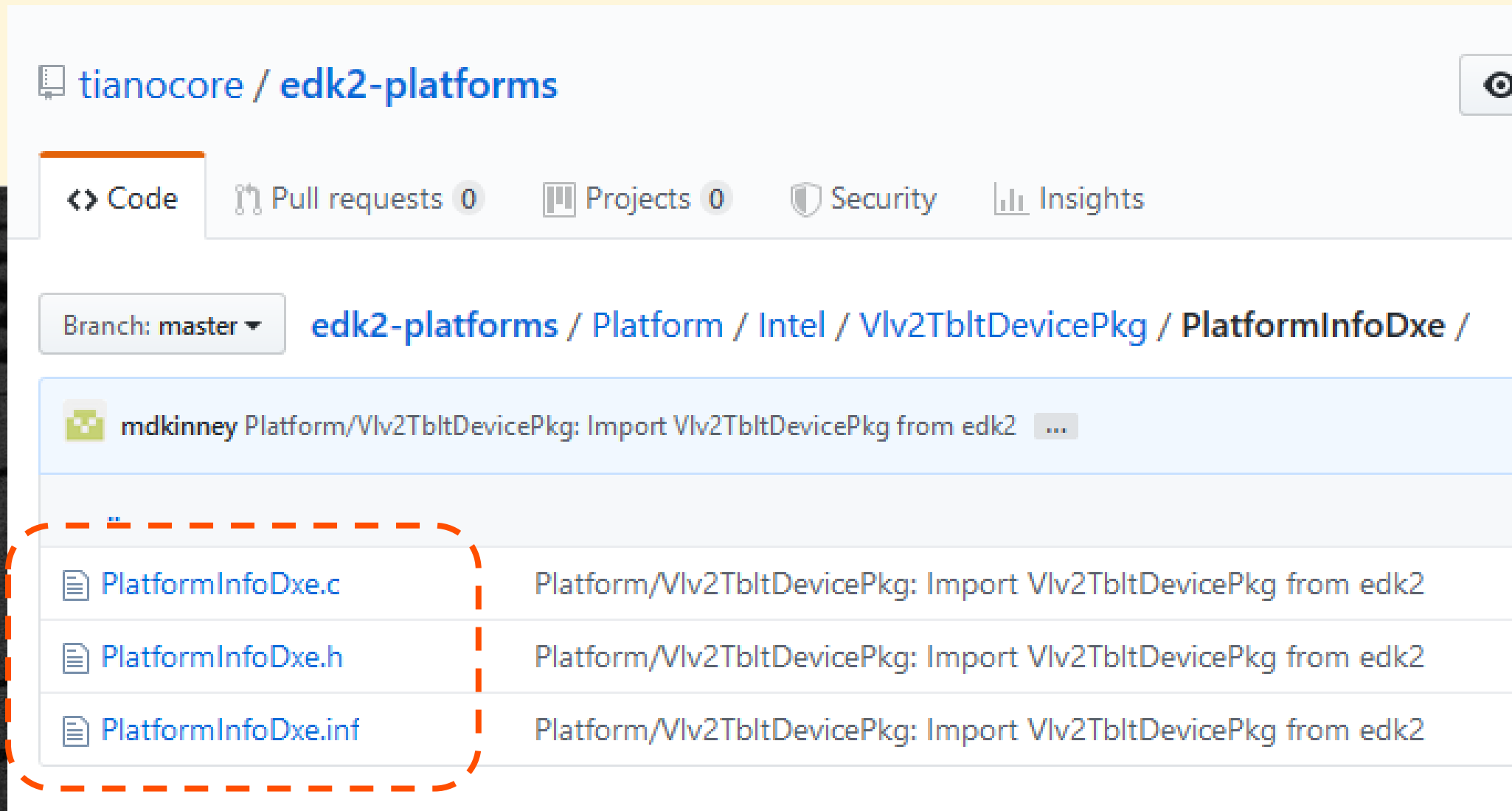
[Protocols]

```
gEfiDiskIoProtocolGuid ## BY_START
gEfiDiskIo2ProtocolGuid ## BY_START
```


DXE Driver Example - PlatformInfoDxe



<https://github.com/tianocore/edk2-platforms/. . ./Vlv2TbItDevicePkg/PlatformInfoDxe>



The screenshot shows the GitHub repository page for `PlatformInfoDxe` within the `edk2-platforms` repository. The breadcrumb navigation path is `edk2-platforms / Platform / Intel / Vlv2TbItDevicePkg / PlatformInfoDxe /`. A commit by `mdkinney` is shown with the message `Platform/Vlv2TbItDevicePkg: Import Vlv2TbItDevicePkg from edk2`. Below the commit, a table lists the files in the directory, which are highlighted with a red dashed box:

<code>PlatformInfoDxe.c</code>	Platform/Vlv2TbItDevicePkg: Import Vlv2TbItDevicePkg from edk2
<code>PlatformInfoDxe.h</code>	Platform/Vlv2TbItDevicePkg: Import Vlv2TbItDevicePkg from edk2
<code>PlatformInfoDxe.inf</code>	Platform/Vlv2TbItDevicePkg: Import Vlv2TbItDevicePkg from edk2

DXE Driver Example – PlatformInfoDxe

 <https://github.com/tianocore/edk2-platforms/PlatformInfoDxe>

Entry Point

“C” File

```
#include "PlatformInfoDxe.h"
...
EFI_STATUS
EFIAPI
PlatformInfoInit (
    IN EFI_HANDLE      ImageHandle,
    IN EFI_SYSTEM_TABLE *SystemTable
)
/*++
{
// ...
return Status;
}
```

INF File

```
[Defines]
...
MODULE_TYPE           = DXE_DRIVER
VERSION_STRING        = 1.0
ENTRY_POINT           = PlatformInfoInit
...

[Depex]
gEfiVariableArchProtocolGuid AND
gEfiVariableWriteArchProtocolGuid
```

Notice the MODULE TYPE, C function Entry point and the [Depex] differences in the INF file

PEI Driver (PEIM) Example - CpuloPei



<https://github.com/tianocore/edk2/UefiCpuPkg/CpuloPei>

tianocore / edk2

Code

Pull requests 0

Projects 0

Insights

Branch: master

edk2 / UefiCpuPkg / CpuloPei /

LeoAtAmd and Igao4 UefiCpuPkg: Modify CpuloPei to support new IoLib library

..	
CpuloPei.c	UefiCpuPkg: Modify CpuloPei to support
CpuloPei.h	Code refinement.
CpuloPei.inf	UefiCpuPkg: INF/DEC file updates to EDK
CpuloPei.uni	UefiCpuPkg: Convert all .uni files to utf-8
CpuloPeiExtra.uni	UefiCpuPkg: Convert all .uni files to utf-8

PEI Driver (PEIM) Example – CpuIoPei

 <https://github.com/tianocore/edk2/UefiCpuPkg/CpuIoPei>

Entry Point

“C” File

```
#include "CpuIoPei.h"
//...
EFI_STATUS
EFIAPI
CpuIoInitialize (
    IN EFI_PEI_FILE_HANDLE FileHandle,
    IN CONST EFI_PEI_SERVICES **PeiServices
)
{
    EFI_STATUS Status;
    //...
    return EFI_SUCCESS;
}
```

INF File

```
[Defines]
...
MODULE_TYPE           = PEIM
VERSION_STRING        = 1.0
ENTRY_POINT           = CpuIoInitialize
...

[Depex]
TRUE
```


LESSON OBJECTIVE

- What is a EDK II Module
- Use EDK II libraries to write UEFI apps/drivers
- How to Define a UEFI application
- Differences between UEFI App / Drivers INF file

Questions?



RETURN TO MAIN TRAINING PAGE



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

