

UEFI & EDK II Training

UEFI Driver Wizard Lab

Currently only available in Ubuntu 16.04

tianocore.org

LESSON OBJECTIVE

Linux Ubuntu 16.04 is only supported

Python Version 2.7 and
python-wxgtk V3.0

Non Ubuntu - Continue to [Porting UEFI Driver Lab](#)

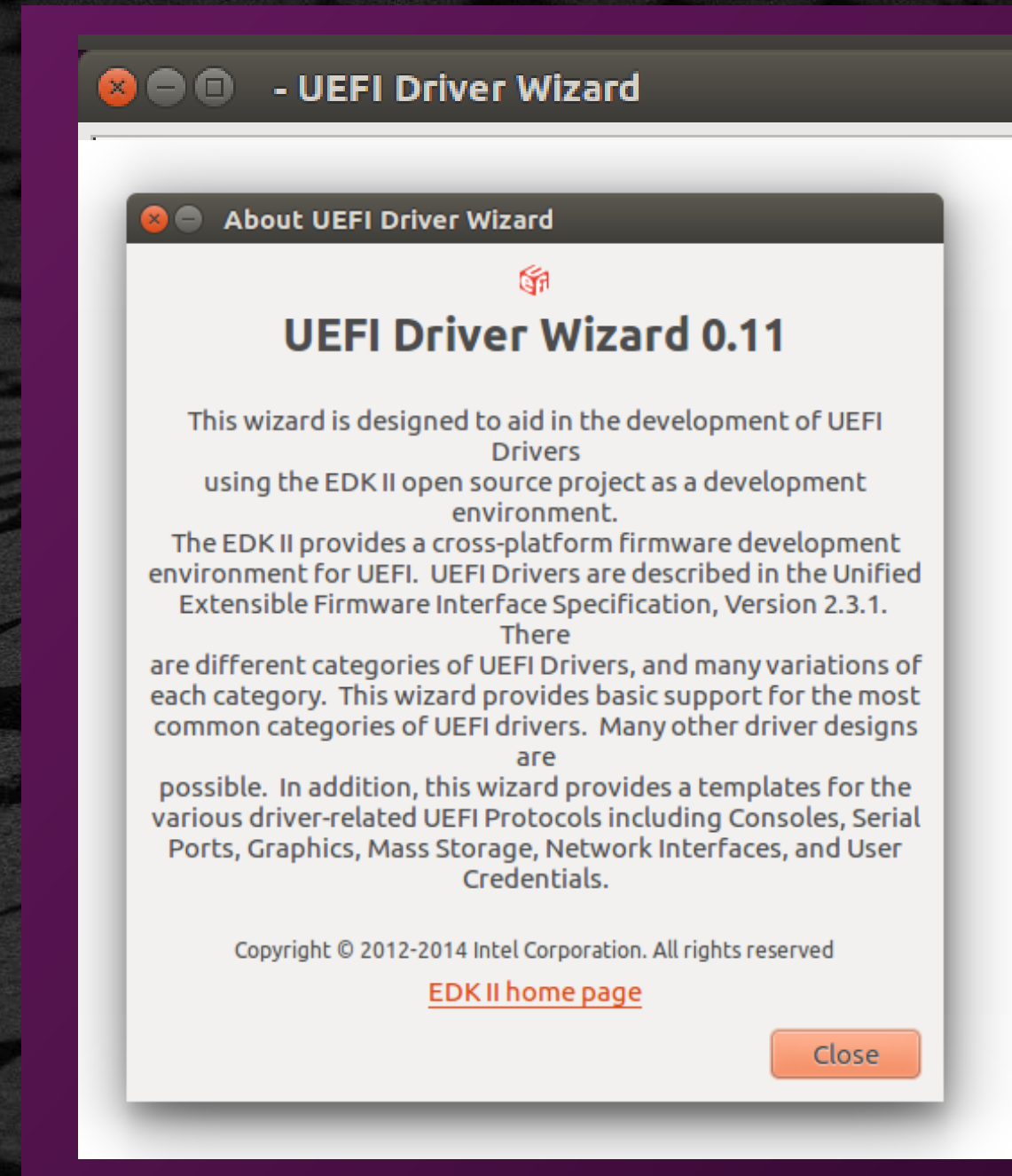
-  Setup the UEFI Driver Wizard
-  Create a UEFI Driver Template

UEFI DRIVER WIZARD

Creating a Template UEFI Driver with the UEFI Driver Wizard

UEFI Driver Wizard Overview

- ✓ Open source tool
- ✓ Based on *Driver Writer's Guide for UEFI 2.3.1* content
- ✓ Intel SSG engineers contributed
- ✓ Located on www.TianoCore.org



Installing Python for UEFI Driver Wizard

Requirements and Options

- Work space must contain BaseTools, MdePkg & MdeModulePkg Packages from [UDK2017](#) for Driver development on Tianocore.org
- Uses previous lab's setup `$HOME/src/edk2`
- Python* scripts from [Github Link](#) then use instructions from README for Python and wxPython versions to install then run

```
bash$ python launch.py
```

Requirements for Your Driver



Using UEFI Driver Wizard

- UEFI Device Driver
- UEFI Version 2.7 (0x00020046)

```
#define EFI_2_70_SYSTEM_TABLE_REVISION ((2<<16) | (70DEC))
```
- Unloadable driver
- Support IA32 & x64 CPUs
- Returns component name information
- Test console device
- Option to produce strings & forms for setup

Template File Contents

Proper UEFI driver entry point

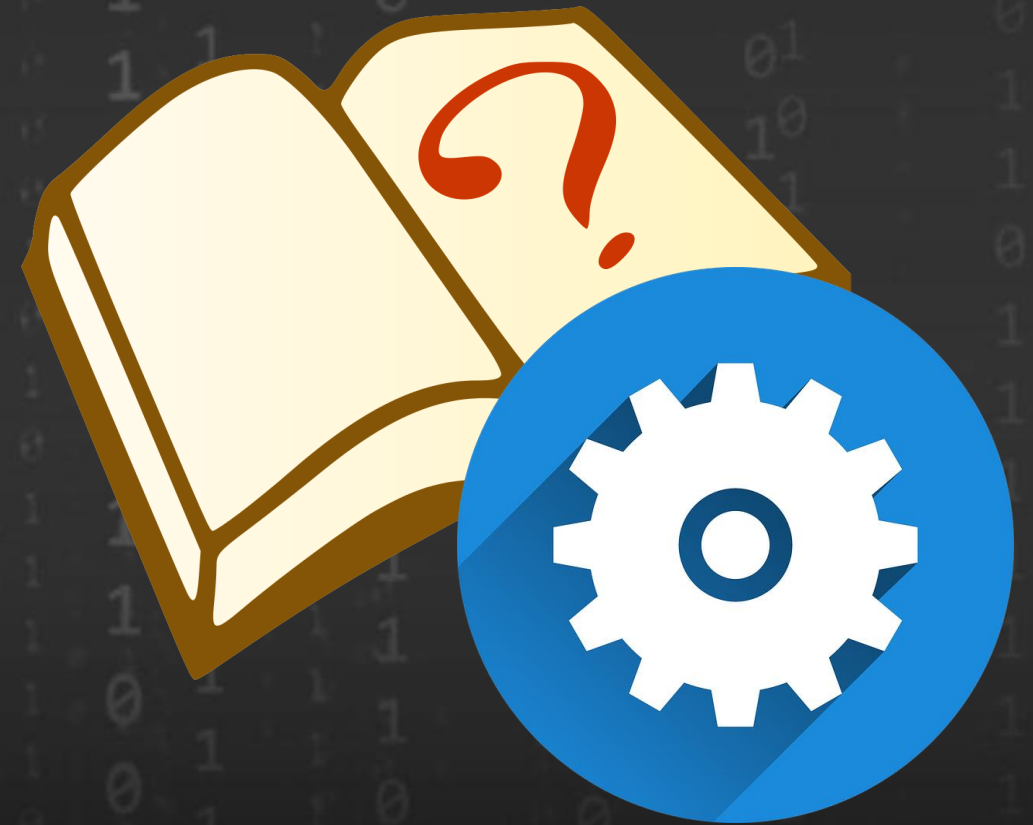
Basic driver libraries/headers

Skeletons for common driver functions

Error values until ported
EFI_UNSUPPORTED, EFI_DEVICE_ERROR

Lab 1: Create a UEFI Driver with the UEFI Driver Wizard

- In this lab, you'll create a new UEFI driver using the UEFI Driver Wizard.
- This will create a set of "c" code files to be used as a template UEFI Driver used in the subsequent driver labs



Lab 1: Install UEFI Driver Wizard, Python & wxPython

1. Perform Lab Setup from previous Labs
2. From the ~FW/DriverWizard folder, copy and paste folder “~FW/DriverWizard/UefiDriverWizard” to ~\$Home
3. Check if version 2.7.x is the default of Python from Terminal Prompt

```
bash$ python -V
```

```
Python 2.7.12
```

3. Install the wxPython (Version 3.0)

```
bash$ sudo apt-get install python-wxgtk3.0
```


Lab 1: UefiDriverWizard -Select Work Space

Terminal Prompt (Cnt-Alt-T)

```
bash$ cd ~UefiDriverWizard
bash$ python launch.py
```

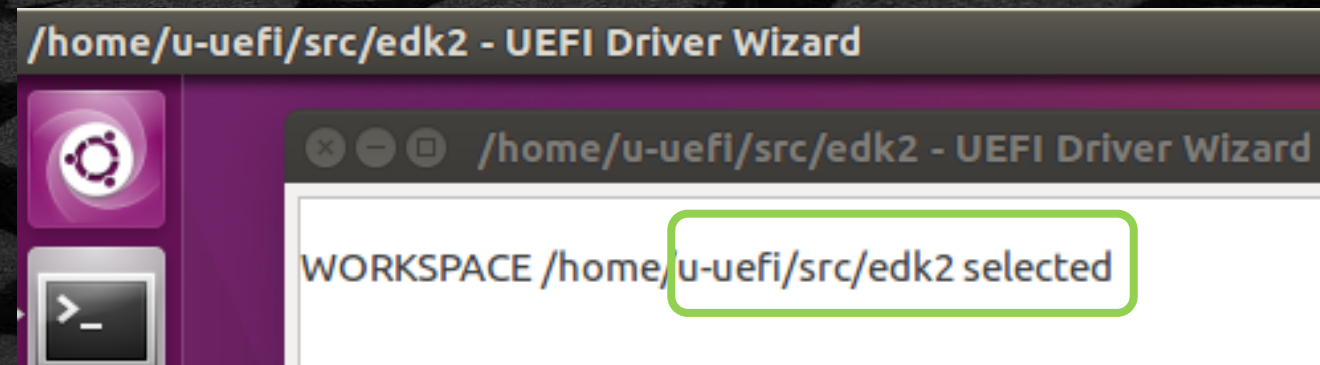
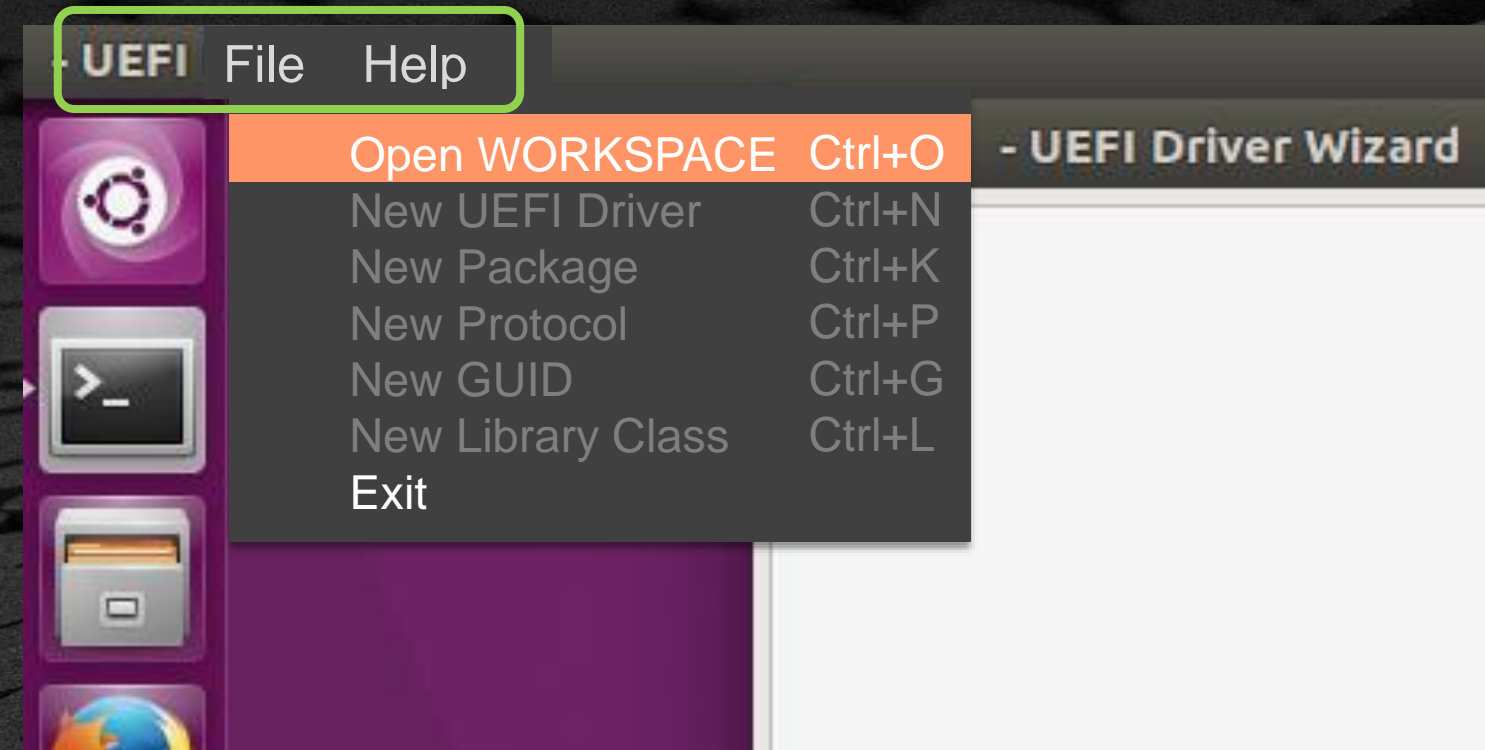
Select a Work Space

Control+O – to browse for a directory

Browse to ~src/edk2

Select

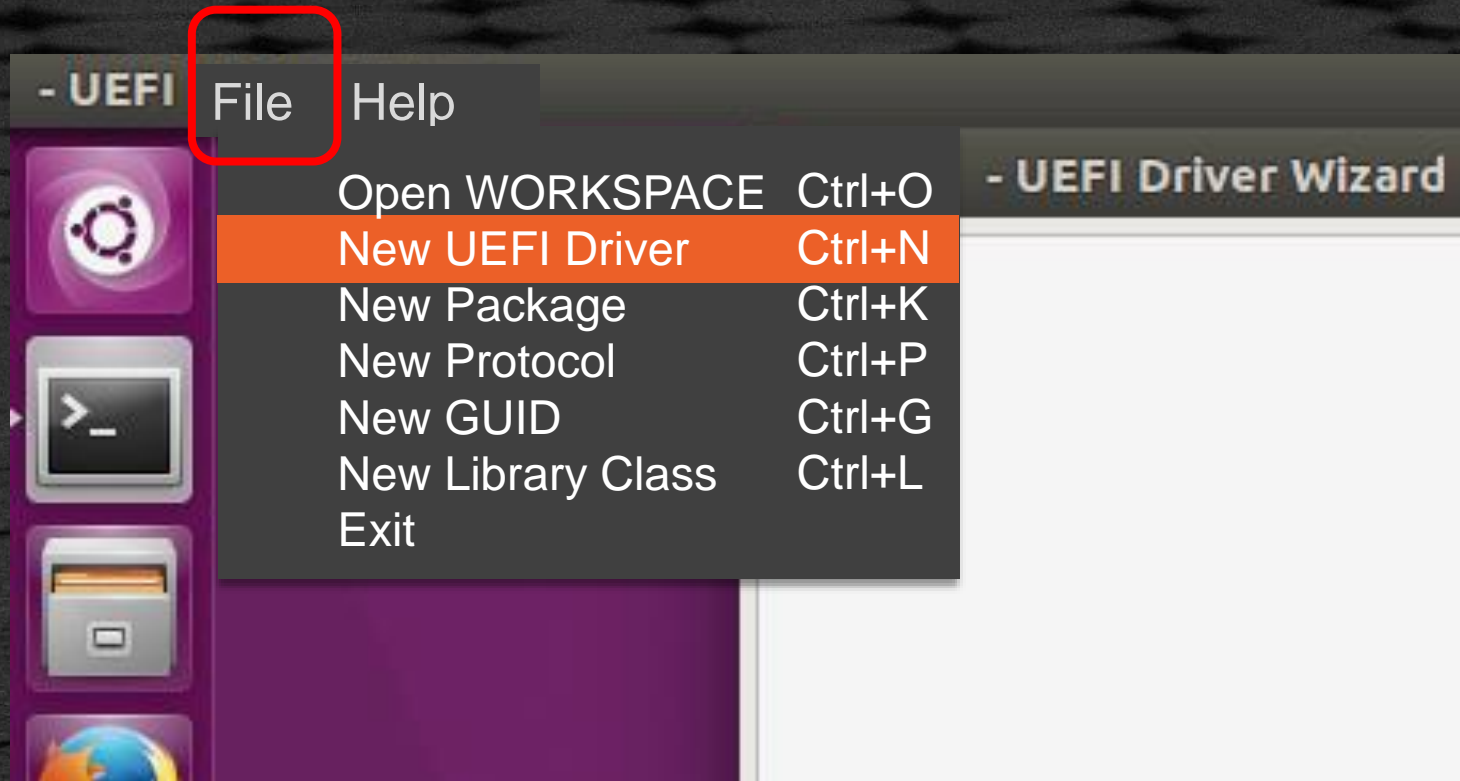
Open



Note: the environment for EDK II must be setup with edksetup.sh

Lab 1: Create a New UEFI Driver

Control+N – to Open Menu



New UEFI Driver

UEFI Driver Path:

UEFI Driver Name:

UEFI Driver Version:

UEFI Driver GUID:

UEFI Driver Type:

- ☒ UEFI Driver Model Device Driver
- ☐ Root Bridge Driver
- ☐ UEFI Driver Model Bus Driver
- ☐ Service Driver
- ☐ UEFI Driver Model Hybrid Driver
- ☐ Initializing Driver

Driver Binding:

Optional Features Common to all UEFI Driver Types:

- ☒ Unloadable
- ☒ Driver Supported EFI Version Protocol
- ☒ HII Packages for Strings, Fonts, or Images

UEFI Specification Version:

CPU Architectures:

- ☐ All CPU Architectures
- ☒ IA32
- ☒ X64
- ☐ IPF
- ☐ EBC

<< Prev **Next >>** Finish Cancel

Lab 1: New UEFI Driver Menu

- UEFI Driver Path” – Type: “MyWizardDriver”

Note: “UEFI Driver Name” is filled in.

- **Ensure** all the forms, radio buttons, and boxes are filled in and **selected exactly** like the image to the right.

- **Note:** A new, specific driver GUID will populate, so it will be different than this image

Click

Next >>

New UEFI Driver

UEFI Driver Path Browse

UEFI Driver Name

UEFI Driver Version

UEFI Driver GUID Generate GUID

UEFI Driver Type

☒ UEFI Driver Model Device Driver ☐ Root Bridge Driver

☐ UEFI Driver Model Bus Driver ☐ Service Driver

☐ UEFI Driver Model Hybrid Driver ☐ Initializing Driver

Driver Binding

Optional Features Common to all UEFI Driver Types

☒ Unloadable

☒ Driver Supported EFI Version Protocol

☒ HII Packages for Strings, Fonts, or Images

UEFI Specification Version

CPU Architectures

☐ All CPU Architectures

☒ IA32

☒ X64

☐ IPF

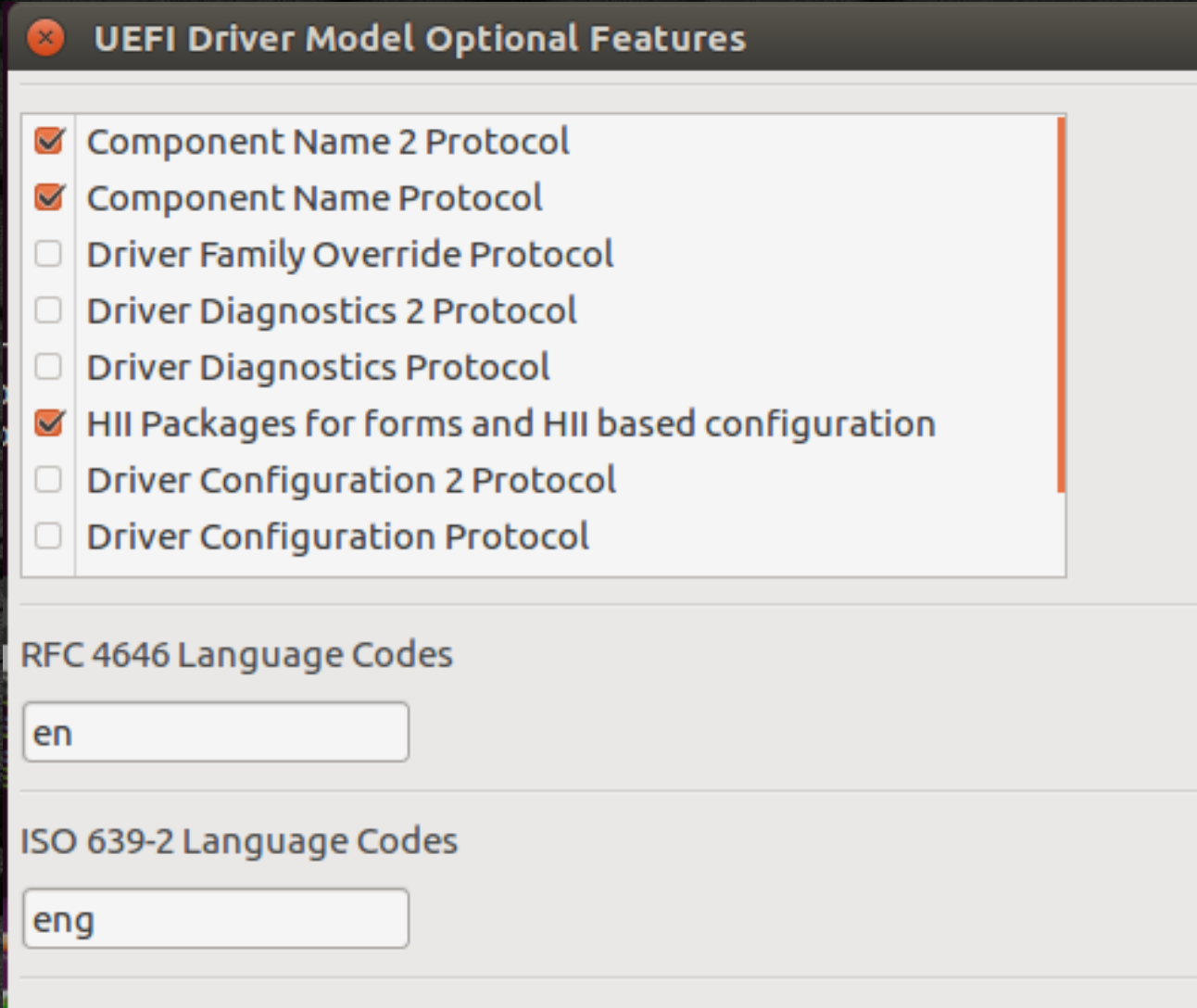
☐ EBC

<< Prev Next >> Finish Cancel

Lab 1: UEFI Driver Model Optional Features

Ensure all the forms, radio buttons, and boxes are filled in and **selected** *exactly* like the image to the right.

- ✓ "Component Name 2 Protocol"
- ✓ "Component Name Protocol"
- ✓ "HII Packages for Forms . . ."



The screenshot shows a window titled "UEFI Driver Model Optional Features". It contains a list of optional features with checkboxes. The following features are checked: "Component Name 2 Protocol", "Component Name Protocol", and "HII Packages for forms and HII based configuration". Below the list, there are two sections for language codes: "RFC 4646 Language Codes" with a text box containing "en", and "ISO 639-2 Language Codes" with a text box containing "eng".

Feature	Selected
Component Name 2 Protocol	Yes
Component Name Protocol	Yes
Driver Family Override Protocol	No
Driver Diagnostics 2 Protocol	No
Driver Diagnostics Protocol	No
HII Packages for forms and HII based configuration	Yes
Driver Configuration 2 Protocol	No
Driver Configuration Protocol	No

RFC 4646 Language Codes

en

ISO 639-2 Language Codes

eng

Click

Next >>

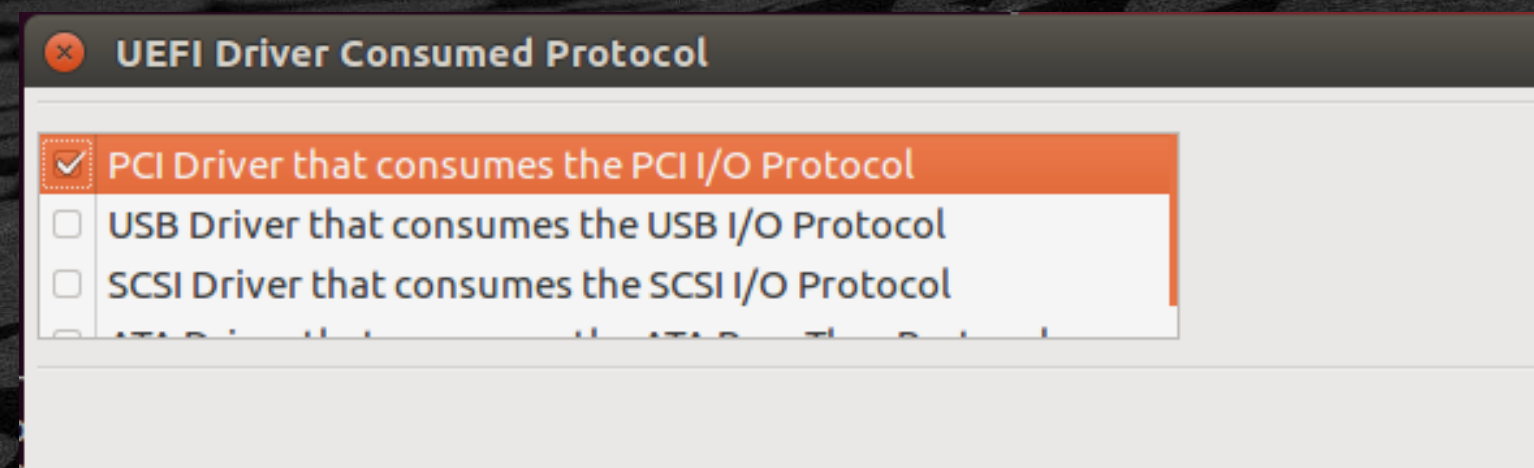
Lab 1: UEFI Driver Consumed Protocol

Select

✓ “PCI Driver that consumes the PCI I/O Protocol”

Click

Next >>



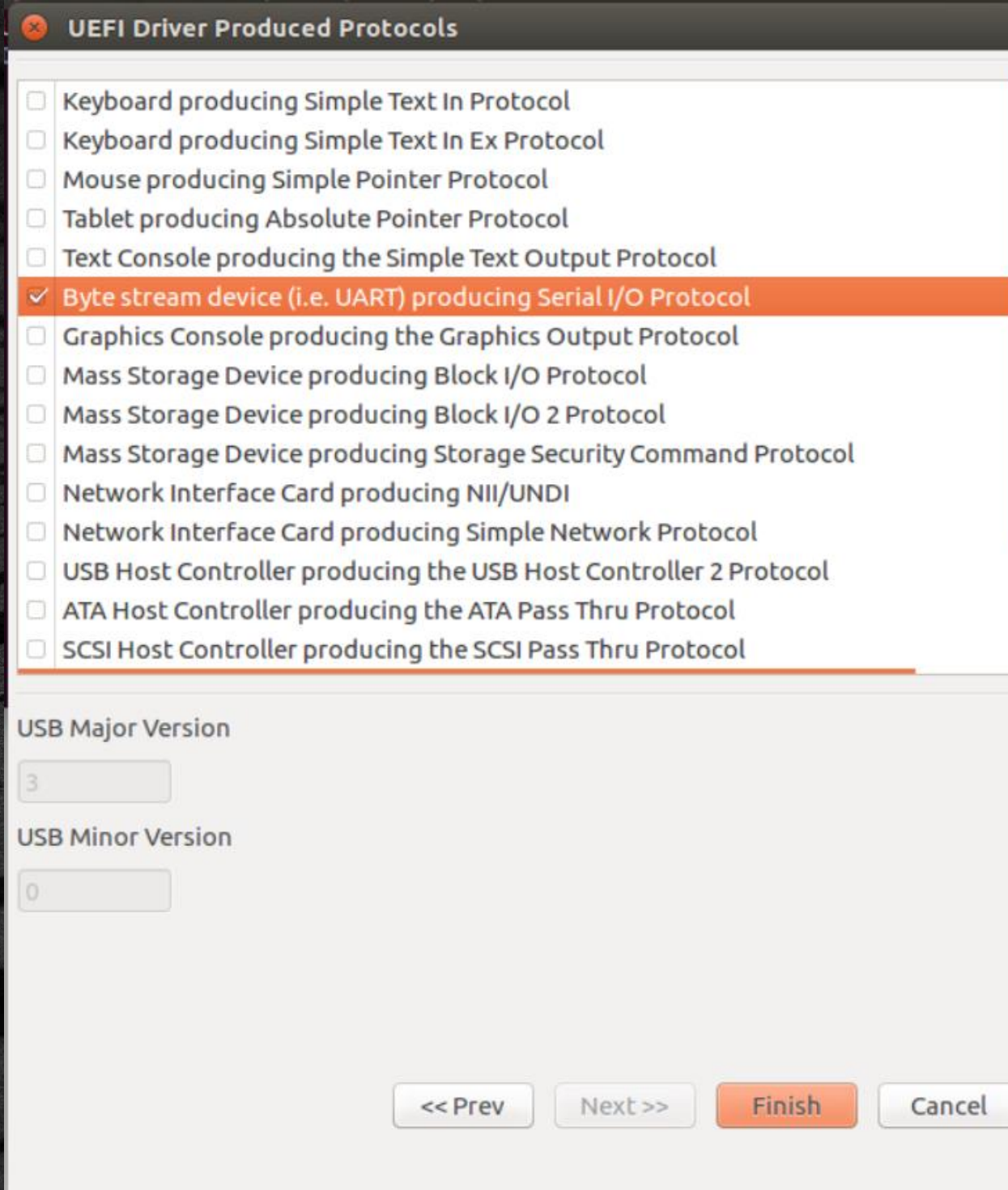
Lab1: UEFI Driver Produced Protocols

Select

✓ "Byte stream device (i.e. UART) producing Serial I/O Protocol"

Click

Finish



UEFI Driver Produced Protocols

- ☐ Keyboard producing Simple Text In Protocol
- ☐ Keyboard producing Simple Text In Ex Protocol
- ☐ Mouse producing Simple Pointer Protocol
- ☐ Tablet producing Absolute Pointer Protocol
- ☐ Text Console producing the Simple Text Output Protocol
- ☒ Byte stream device (i.e. UART) producing Serial I/O Protocol
- ☐ Graphics Console producing the Graphics Output Protocol
- ☐ Mass Storage Device producing Block I/O Protocol
- ☐ Mass Storage Device producing Block I/O 2 Protocol
- ☐ Mass Storage Device producing Storage Security Command Protocol
- ☐ Network Interface Card producing NII/UNDI
- ☐ Network Interface Card producing Simple Network Protocol
- ☐ USB Host Controller producing the USB Host Controller 2 Protocol
- ☐ ATA Host Controller producing the ATA Pass Thru Protocol
- ☐ SCSI Host Controller producing the SCSI Pass Thru Protocol

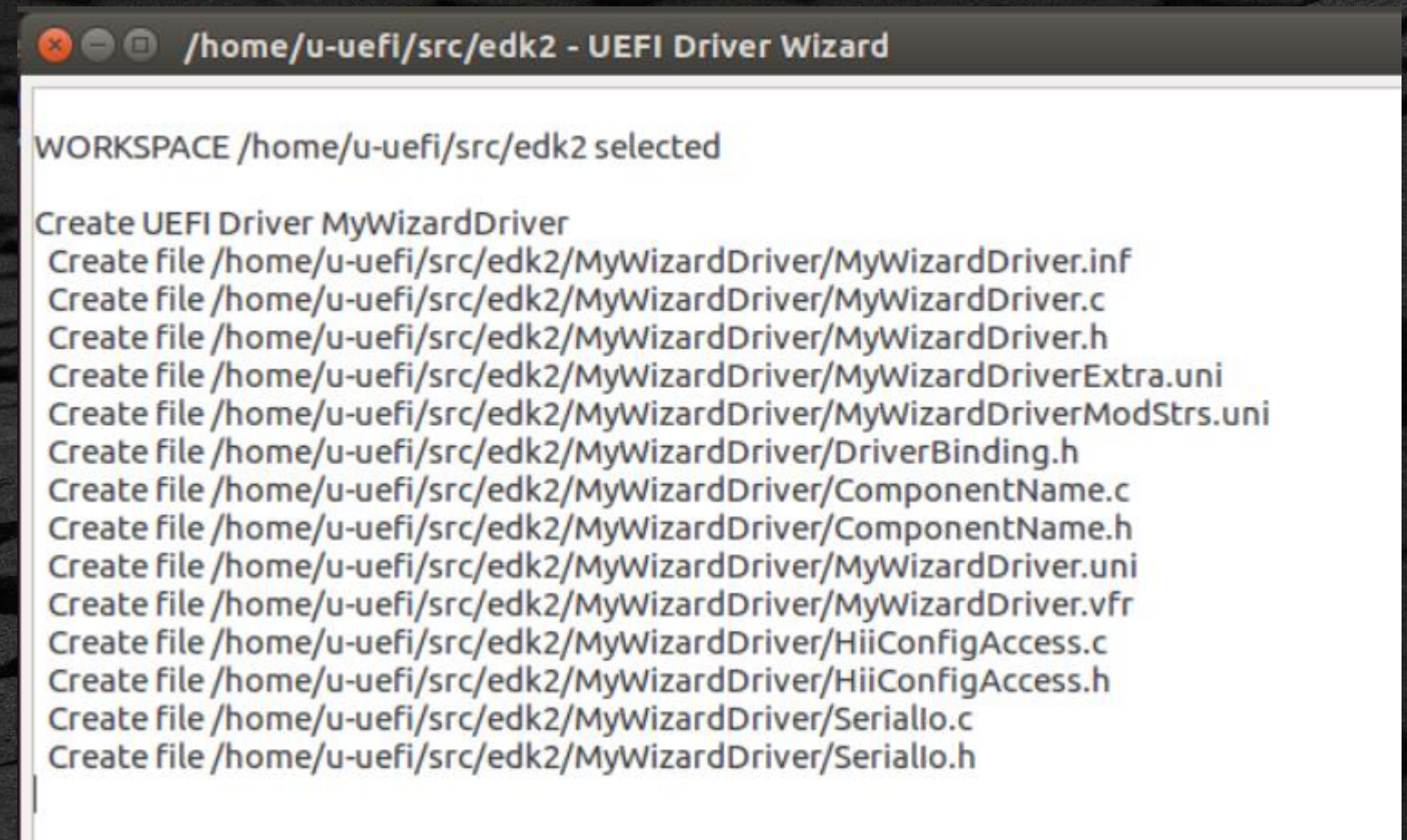
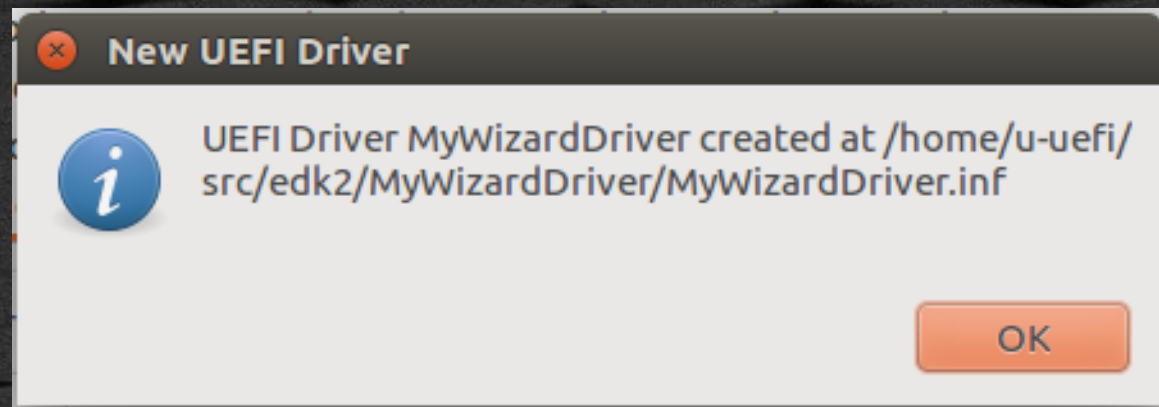
USB Major Version
3

USB Minor Version
0

<< Prev Next >> Finish Cancel

Lab 1: UEFI Driver Created

UEFI Driver template created



SUMMARY

- ✿ Setup the UEFI Driver Wizard
- ✿ Create a UEFI Driver Template

Questions?

