Topic 4

Software Design

Part 1

Outline

- Intro to software design
- Modelling Object-Oriented Systems
- OO Design Principles Intro

Design

“Weeks of coding can save you hours of design”

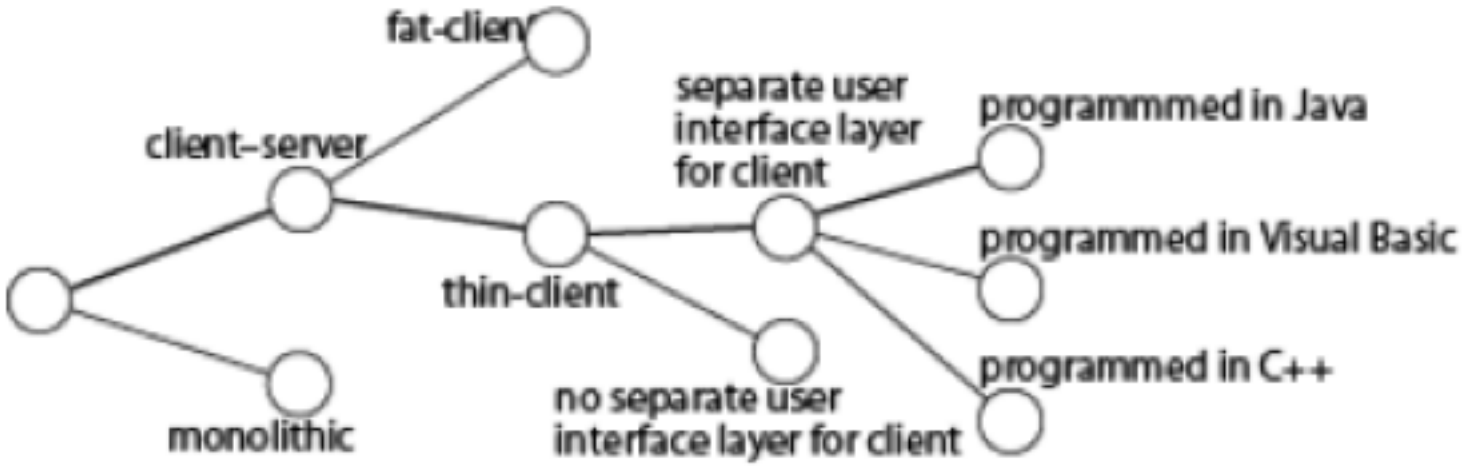
Design

A problem-solving process whose objective is to find and describe a way:

- To implement the system's **functional requirements**
- Respecting constraints imposed by ***non-functional requirements***
- Adhering to general principles of good ***quality***

Design Space

Set of all possible designs that could be achieved with different combinations of design decisions



Design Levels

- Software Architecture
- Component Level
  - Object Oriented Design

Software Architecture

- High level overview of the system
  - System: a logical entity, having a set of definable responsibilities or objectives and consisting of hardware, software or both
- Divide the system into
  - subsystems
    - and their components
- Define how these will interact

Component

- Any piece of software or hardware with a clear role
  - each can be isolated (and therefore replaced)
  - may be reusable or special-purpose

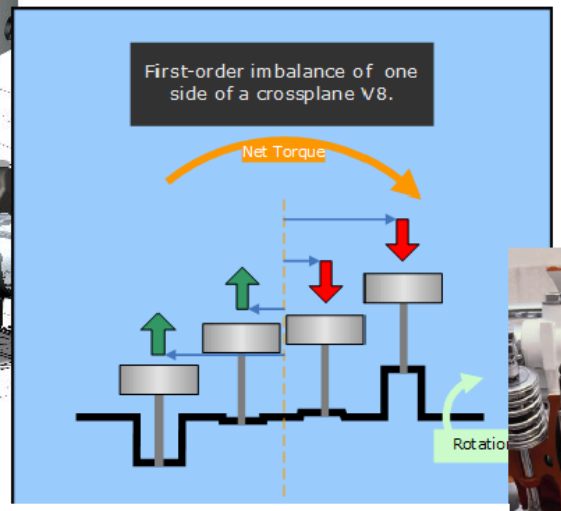
Architecture Strategies

- Pipes and filters
  - Unix commands
- Event driven
  - communication between components through Events
- Client/Server
  - Separate these two main components → typically in networked applications
- Layered
  - System is divided into layers which communicate only with neighboring layers
- Database Centric
  - Separate software components deal only with a central database, not each other

Software Design

- Plan for the modules(packages) and their classes
- Involves Modelling
  - UML diagrams

Modelling - What is this?



Modelling

- Before large undertakings we typically build models
  - blueprints
  - CAD Modelling
- With software, we can model the system or software

Modelling

- Models
  - use standard notation
  - are understandable to clients and users
  - provide abstraction
- Used to
  - aid in design
  - permit and facilitate analysis/review of design
  - can become core documentation fro the system

Ex. UML Diagrams

- Class Diagrams
  - Most widely used
  - Describe classes and their relationships
- Interaction diagrams
  - shows how objects interact with eachother
- State diagrams
  - shows the behaviour of the system internally
- Component/Deployment diagrams
  - show how the various components of systems are arranged logically and physically

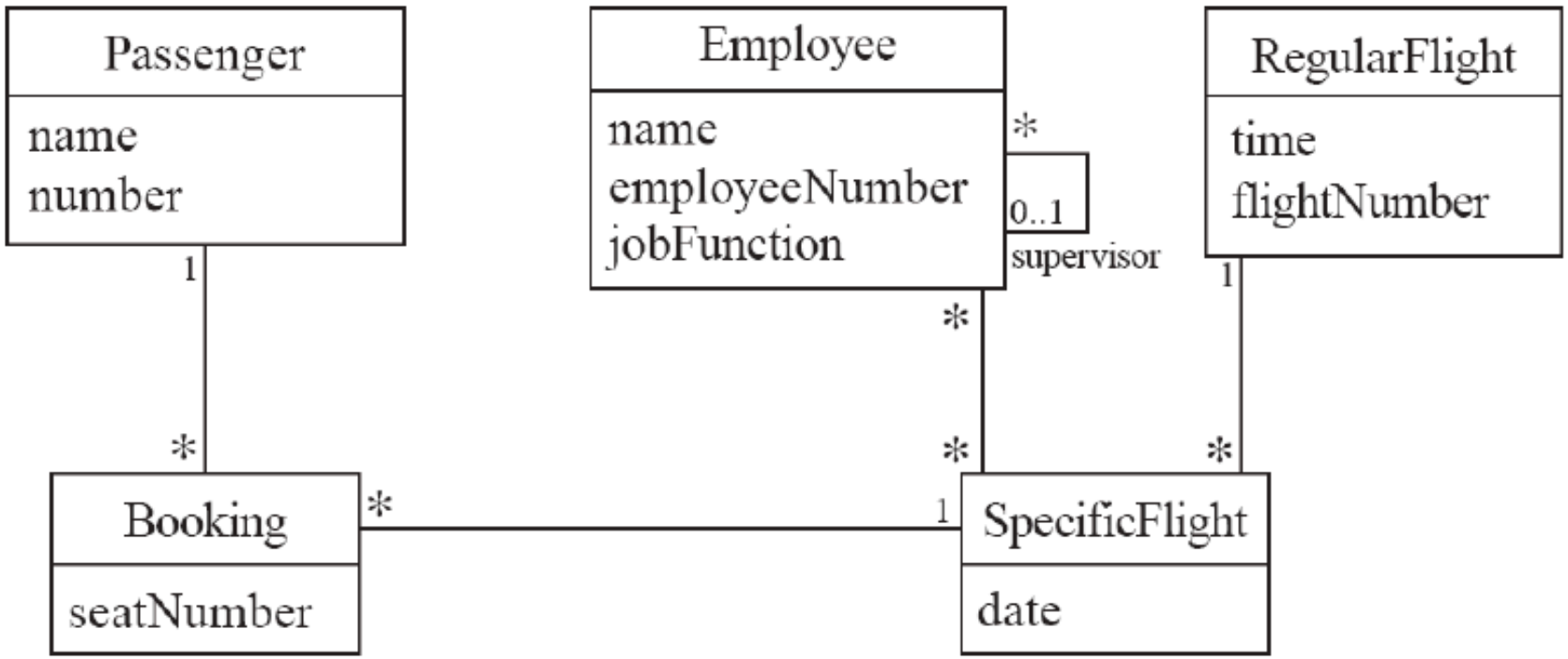
Object-Oriented (OO) Design

- Identifying classes
  - Look at domain / requirements
    - Extract nouns and nouns phrases
    - Eliminate those that are:
      - redundant
      - represent instances (rather than potential classes)
      - too vague (but these may be good generalizations)
    - Extract actions
      - related actions may be responsibility of a single class

OO Design

- Identifying attributes
  - look for the information that must be maintained about each class
  - some nouns rejected as classes will become attributes
  - generally a simple value (int, String, etc)

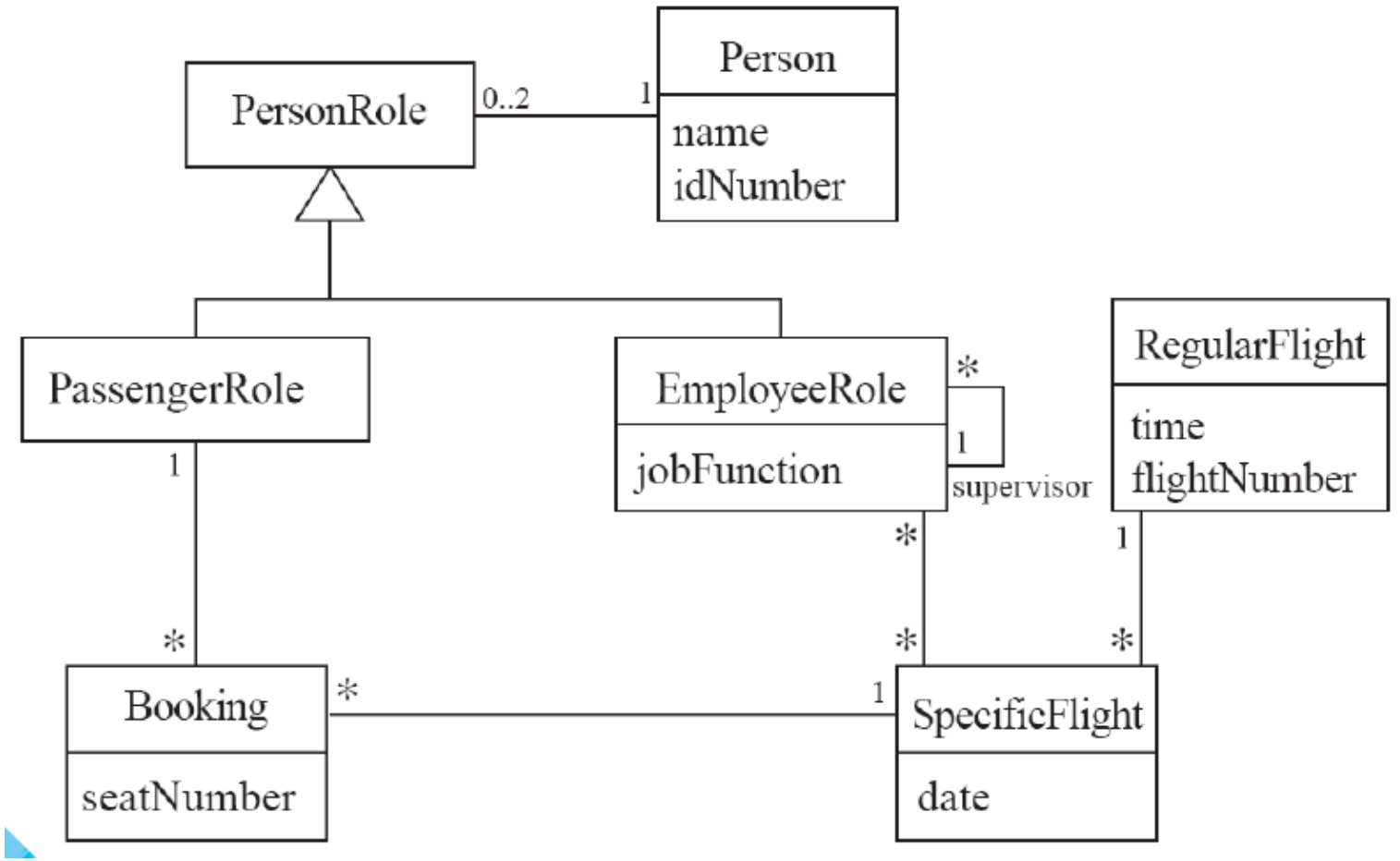
Ex. Attributes



OO Design

- Identifying generalizations (inheritance) and interfaces
  - bottom up
    - group together similar classes and create superclass
    - pull up attributes
  - top down
    - keep classes general at first
    - specialize to subtypes if needed
- Choose interface over inheritance:
  - when classes are dissimilar in attributes, but similar in operations
  - when one or more of them already have a superclass
  - it is clear multiple implementations of the same class could be made

Ex. Generalization



OO Design

- Identifying responsibilities
  - A responsibility is something that the system is required to do.
  - Each functional requirement must be attributed to one of the classes
    - If a class has too many, consider splitting it into distinct classes
    - If a class has none then it is probably useless
  - When a responsibility cannot be attributed to any of the existing classes, then a new class should be created
- To determine responsibilities
  - Analyze your user stories
  - Look for verbs and nouns describing actions in the system description

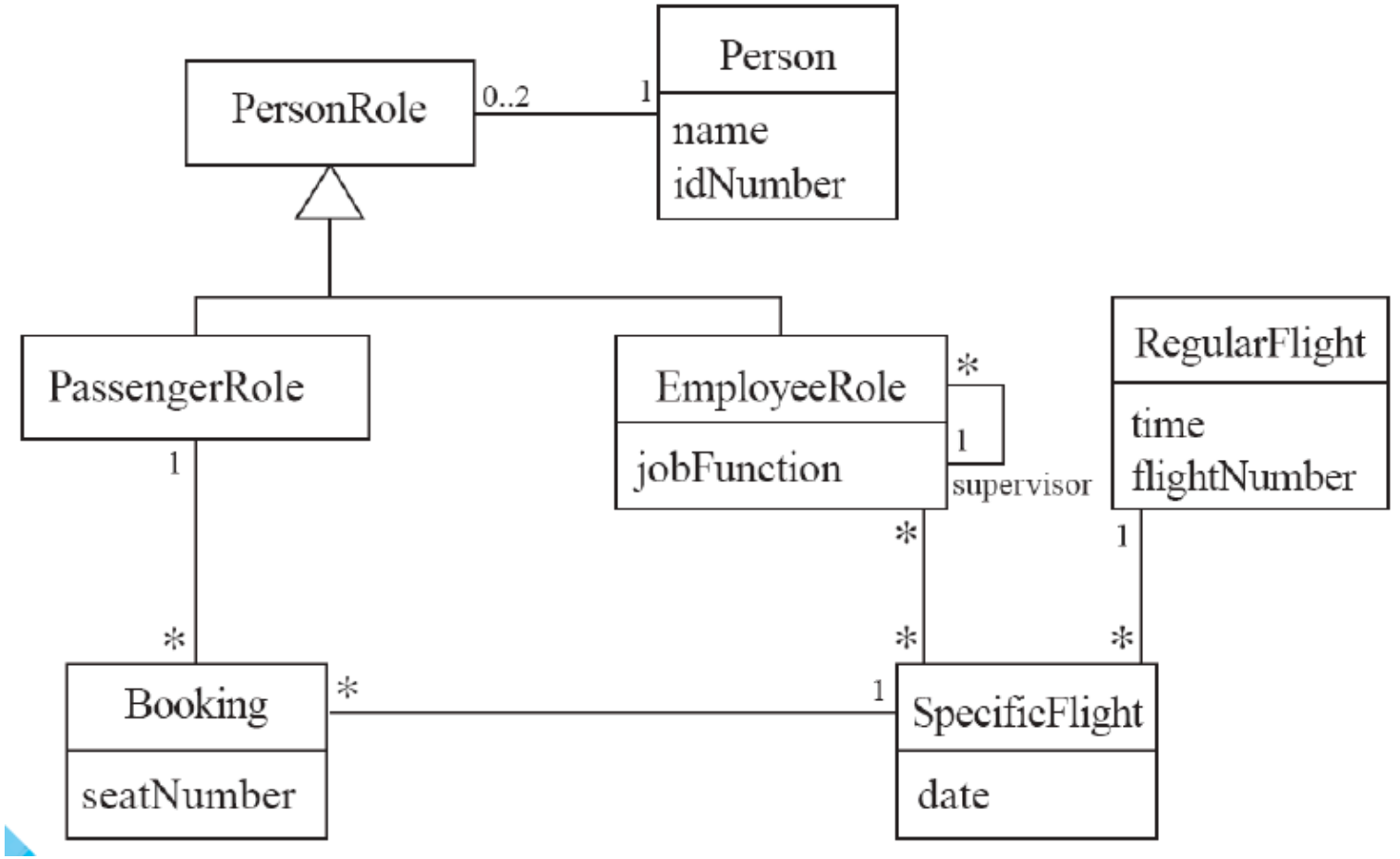
OO Design

- Identifying operations
  - The methods we will use to realize the responsibilities
    - Setting and getting the values of attributes
    - Creating new instances, destroying instances
    - Loading to and saving from persistent storage
    - Adding and deleting links of associations
    - Copying, converting, transforming, transmitting or outputting
    - Computing numerical results
    - Navigating and searching
    - Other specialized work

Implementing Class Diagrams

Attributes	Instance variables
Generalizations	Inheritance (with `extends`)
Interfaces	Interface (with `implements`)
Associations	Instance variables
- one way	- declare reference variable of that Class
- two way	- essentially two one-way associations
- with multiplicity	- use collection of the other Class type (`List, Vector`)

Ex. Implementing RegularFlight and SpecificFlight association



Ex. The ONE side of a 1-> *

```
class SpecificFlight
{
   private Calendar date;
   private RegularFlight regularFlight;

   // Constructor that should only be called from
   // addSpecificFlight
   SpecificFlight(Calendar date,
                  RegularFlight regularFlight) {
      this.date = date;
      this.regularFlight = regularFlight;
   }
}
```

Ex. The MANY side of 1->*

```
class RegularFlight
{
   private List<SpecificFlight> specificFlights;

   // Method that has primary responsibility
   public void addSpecificFlight(Calendar aDate)
{
      SpecificFlight newSpecificFlight;
      newSpecificFlight = new
                        SpecificFlight(aDate, this);

     specificFlights.add(newSpecificFlight);
   }
}
```

Software Design

- Design Principles
  - are guidelines for coding
  - aim for high quality software
    - reusable, extensible, flexible, etc.

Design Principles - Basics

- Low coupling
- High cohesion

Coupling

- Describes the interdependence of entities
- Entities appear separate
  - but bound together to be functional

Coupling

- High/Strong
  - One object uses internal data of another
  - Sharing global variables

Low/Weak

- Share only data through parameters and returns
- Pass events or messages

Coupling

- Low coupling
  - Information hiding and encapsulation
  - Stable class interfaces

- High coupling
  - Undesirable
  - Difficult to change and extend (and read!)

Cohesion

- intrarelatedness of an entity
  - ie. within a single class
- Relatedness of functionality and/or the data of an entity

Low Cohesion

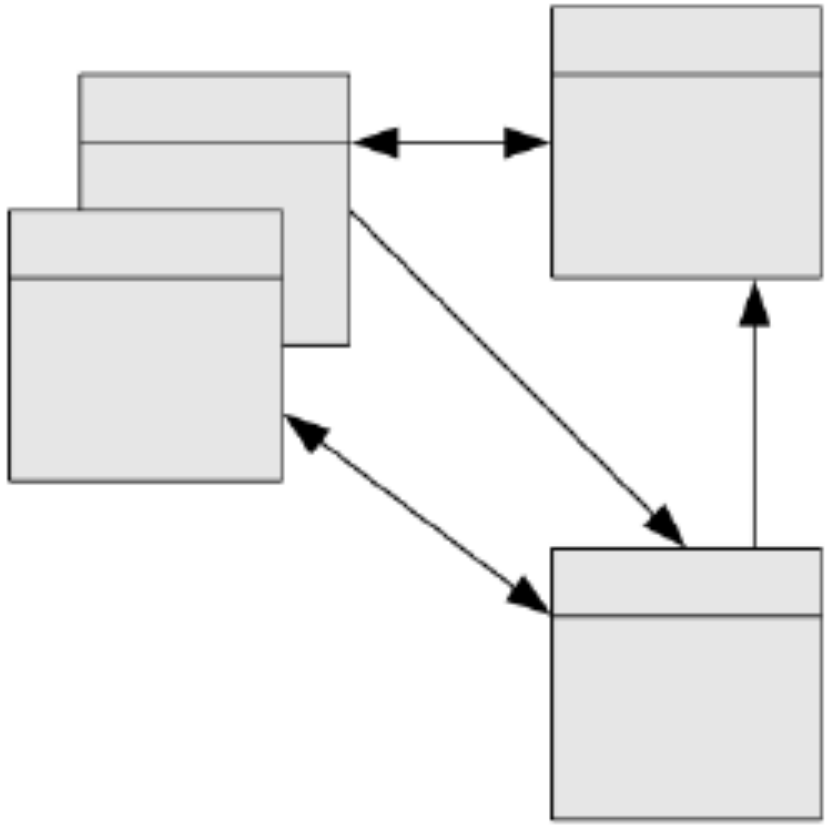
- Related only because they are grouped
- Names often vague
  - "Utilities"
- Groups of functions often act on distinct subsets of data
- "God class"

High cohesion

- Well defined purpose
- Specific names
- Functions act on same set of data
- Improves clarity of entities and their dependencies
- Easier to maintain and extend

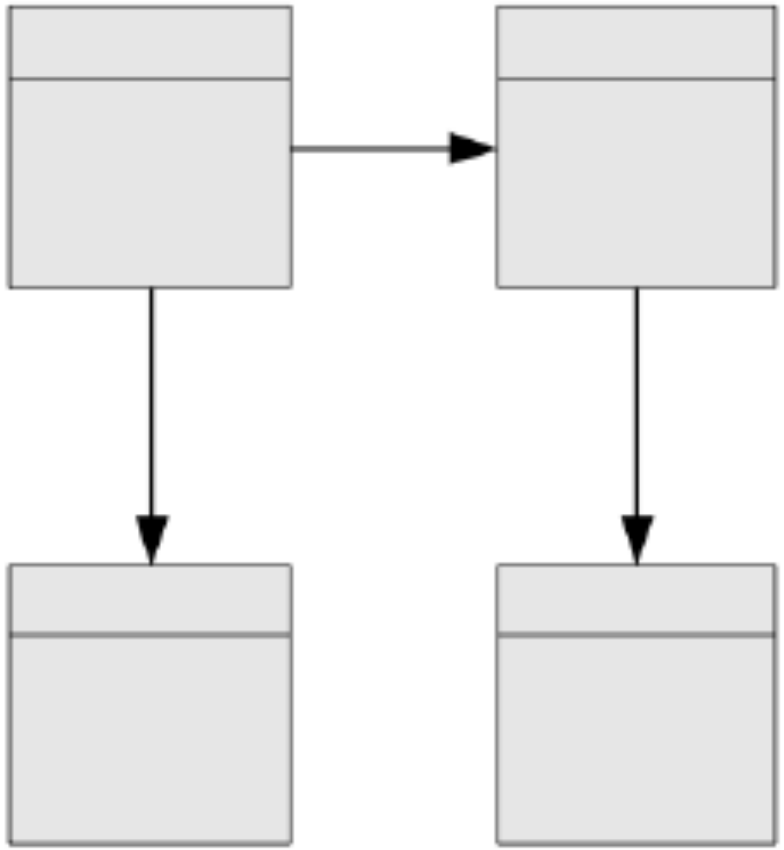
High coupling ←→ Low Cohesion

- Undesirable
- hard to read and understand what does what
- boundaries are weak and few
- unstable to change



Low coupling ←→ High Cohesion

- Often go together
- Entities are responsible for specific functionality
- Entities are more independent



Getting Started with Grails

Flat learning curve

Convention-over-Configuration, sensible defaults, opinionated APIS and the Groovy language combine to make Grails easy to learn for Java developers

http://www.grails.org/

Getting Started with Grails

Using Linux (or MacOS) is highly recommended.

- Outside of the MS ecosystem, most of web applications run on Linux.
- Most tools are easily installed, better supported
- Linux is free, can be installed in VM

Virtual Machine

VirtualBox

- Free to use from Oracle, Open Source
- https://www.virtualbox.org/

VMware

- Commercial software, UWO CS
  students can get a free academic license
- http://www.csd.uwo.ca/help/software/vmap.html

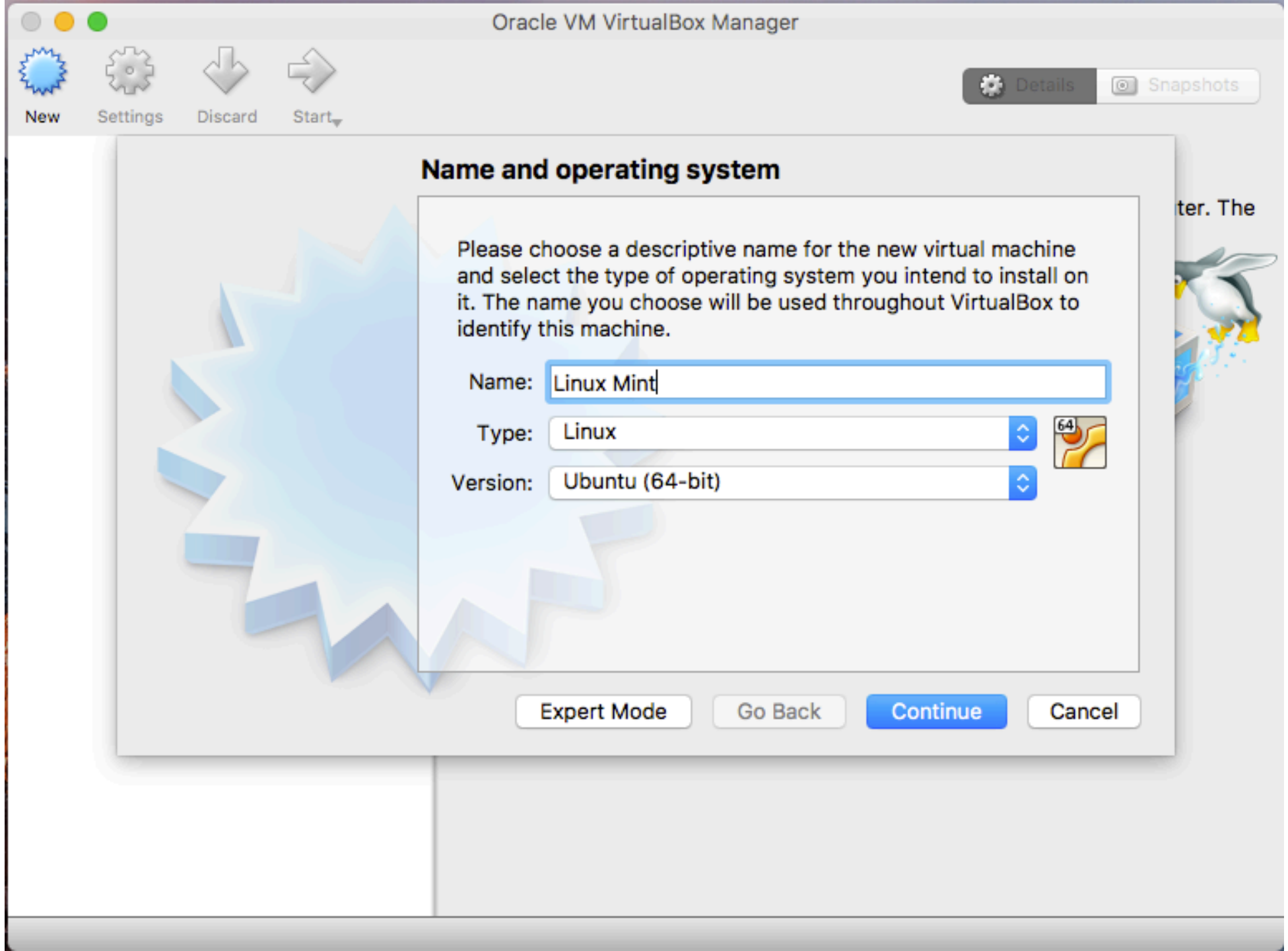
Distributions of Linux

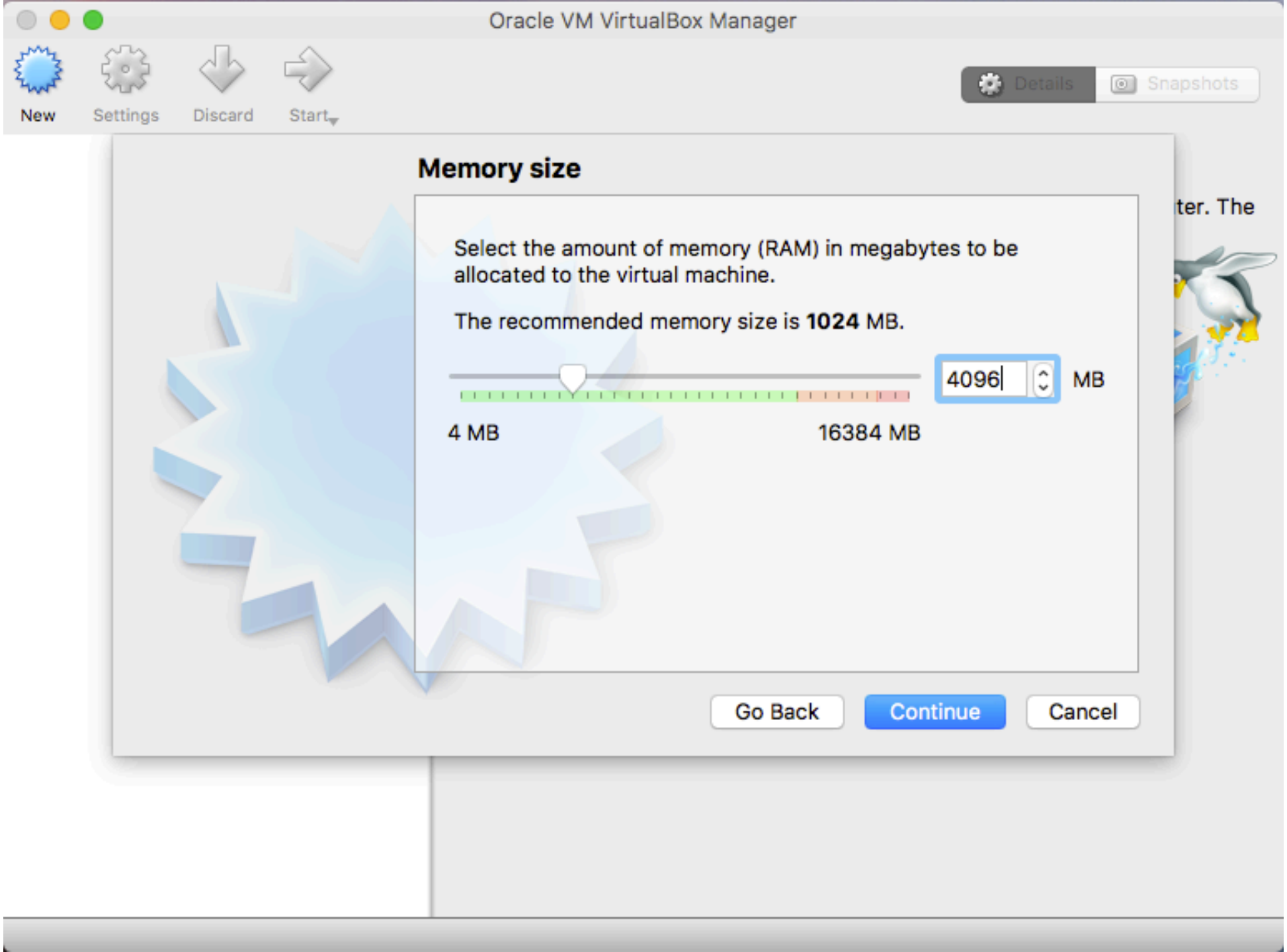
- Too many to name
- I've had good luck with Linux Mint
- https://www.linuxmint.com/

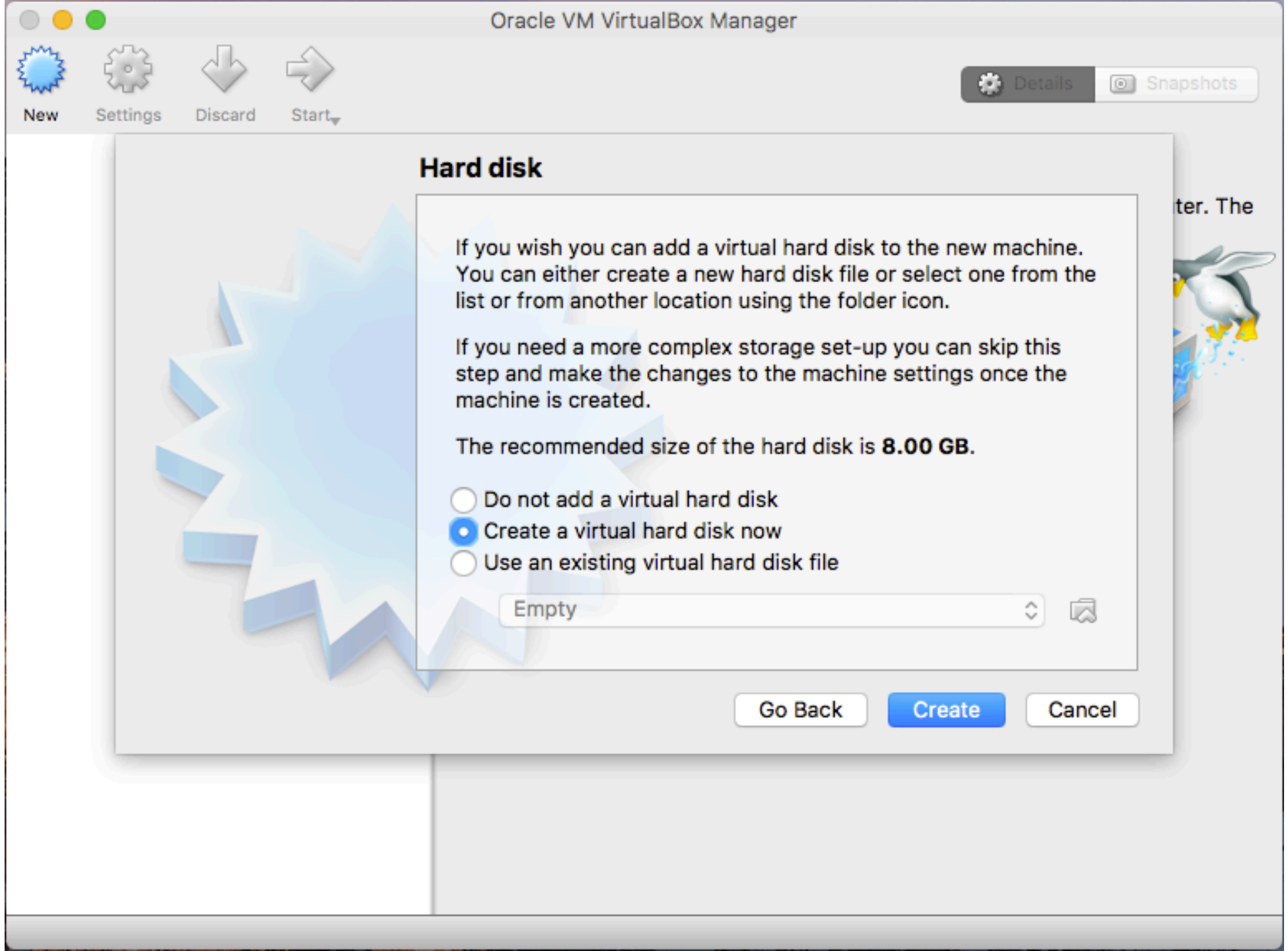
If you haven't used Linux much before, now is a great time to get acquainted.

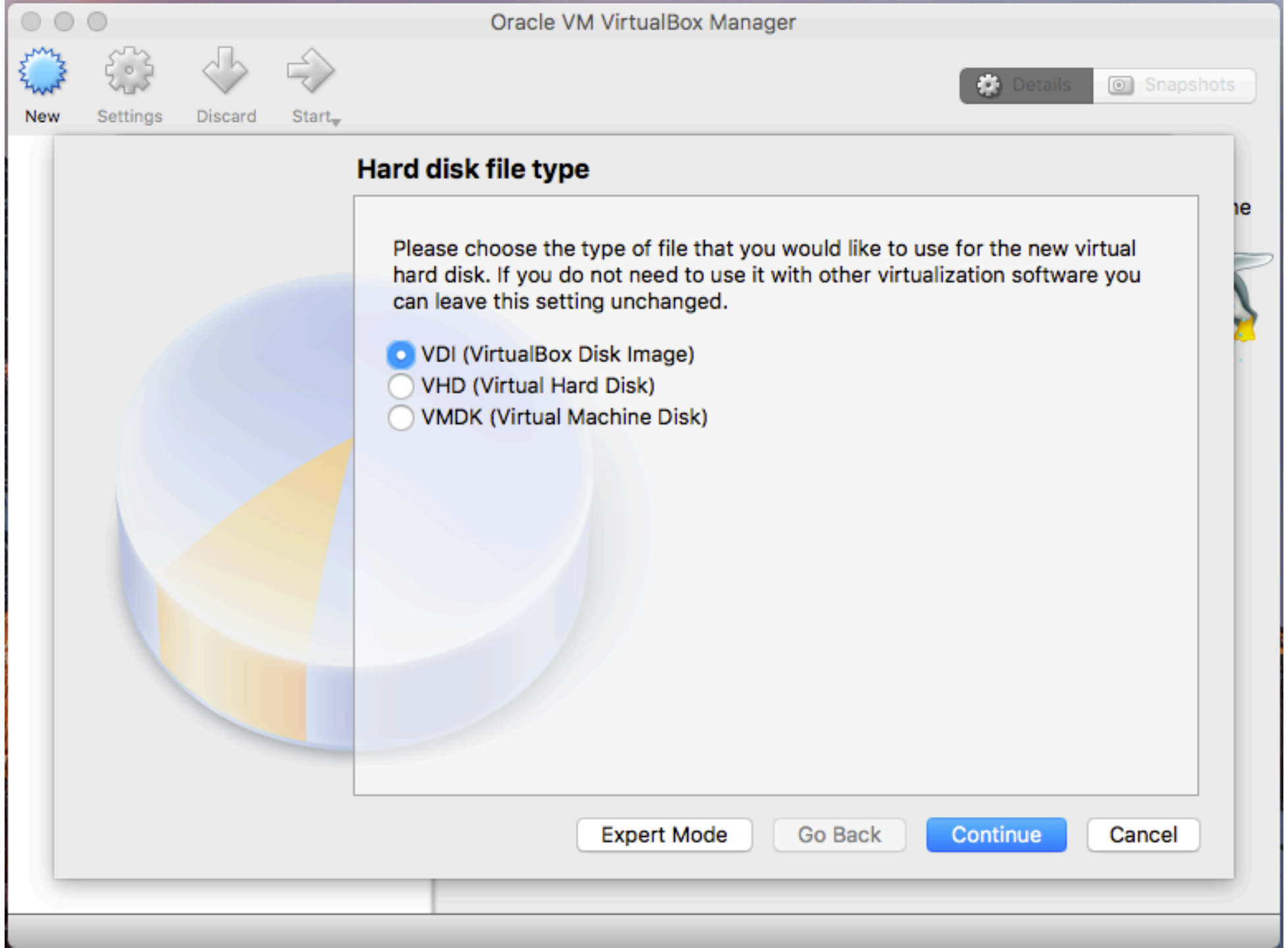
Installing Linux on a Virtual Machine

- Download image file
- https://www.linuxmint.com/download.php
- Select a Desktop Environment and Architecture
  - Cinnamon and 64-bit is probably suitable
- Select a download mirror

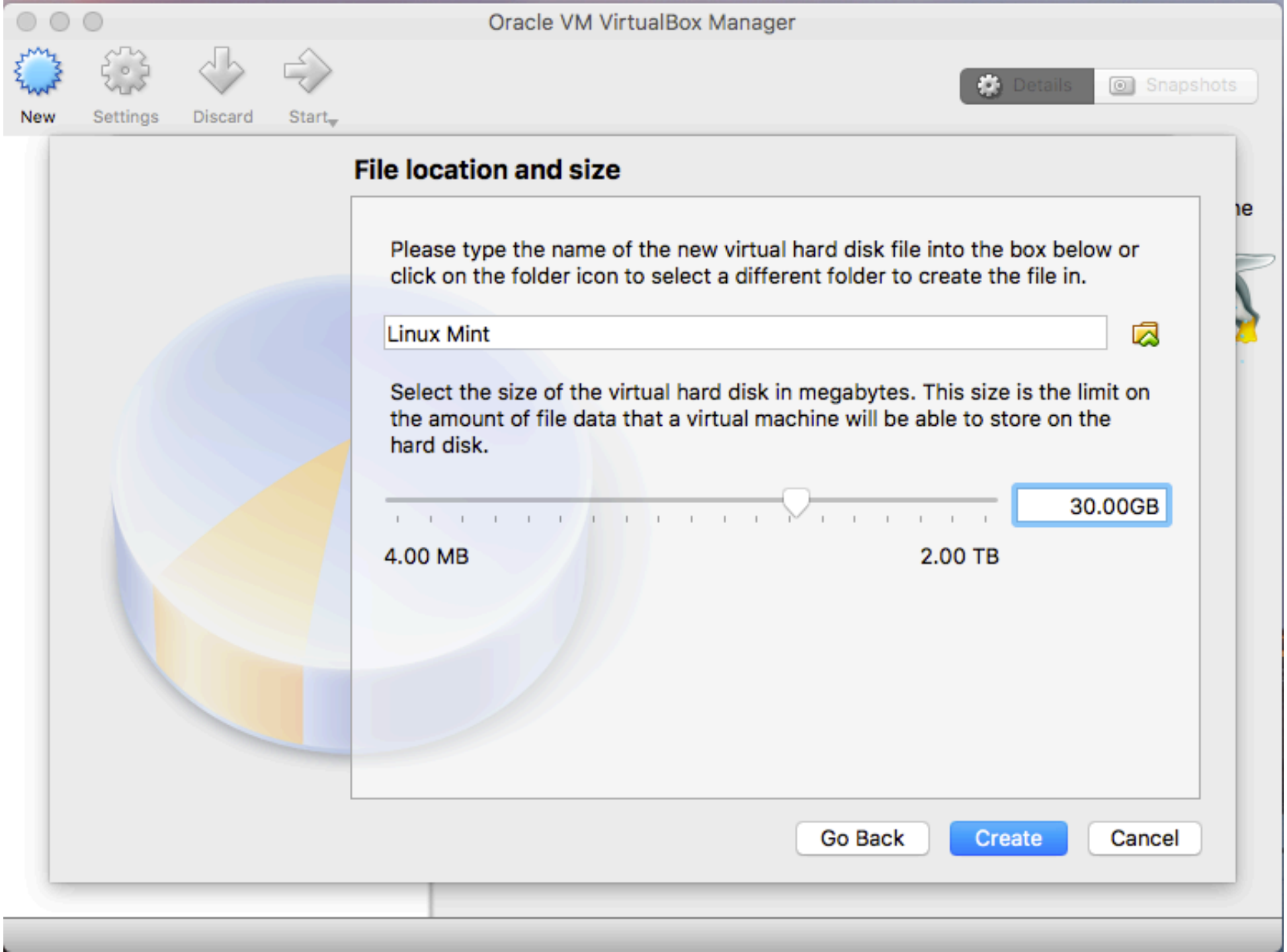

Oracle VM VirtualBox Manager
New
Settings
Discard
Start
Details
Snapshots
Name and operating system
Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.
Name: Linux Mint
Type: Linux
Version: Ubuntu (64-bit)
64
Expert Mode
Go Back
Continue
Cancel

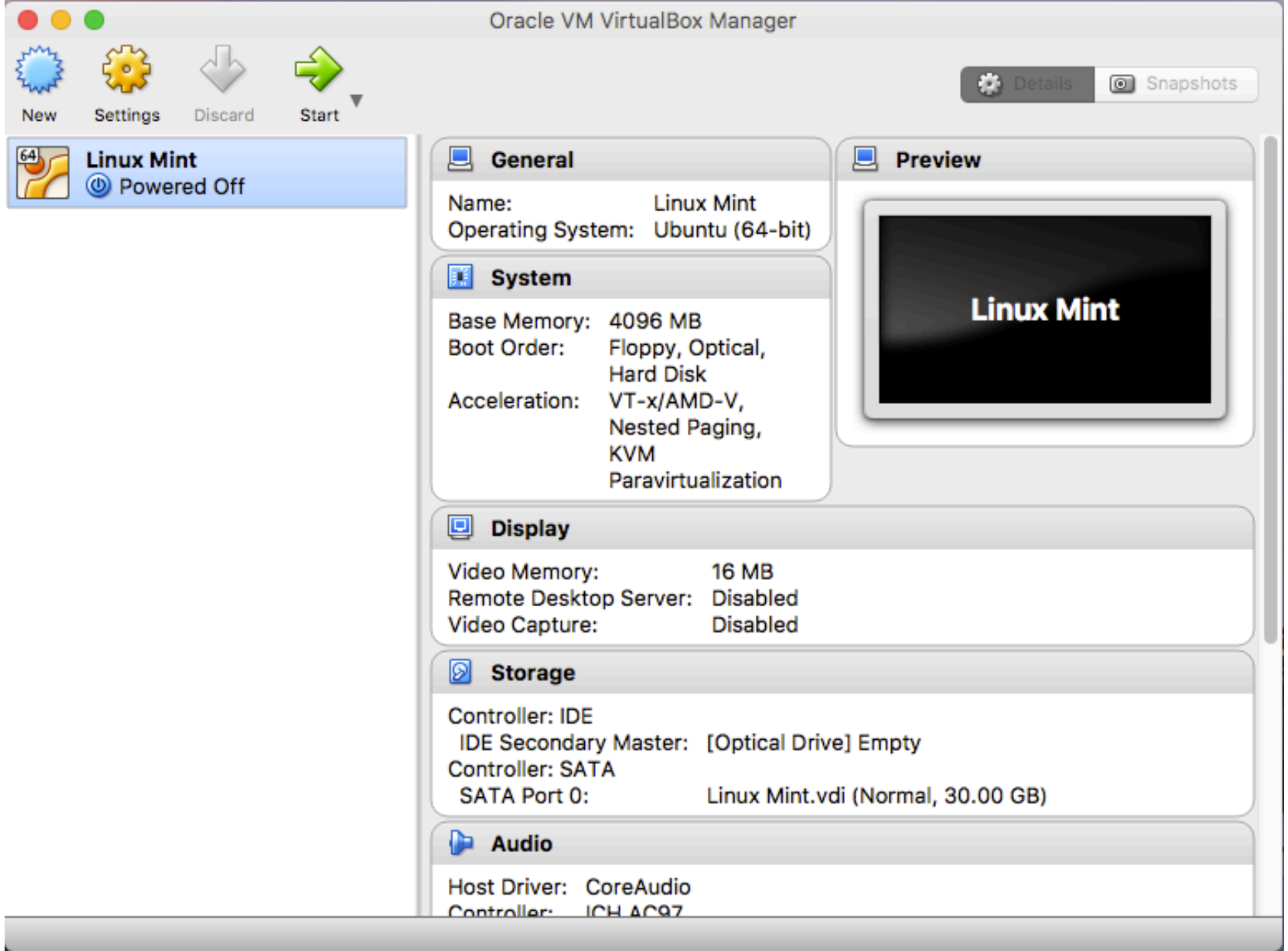

Oracle VM VirtualBox Manager
New
Settings
Discard
Start
Details
Snapshots
Memory size
Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.
The recommended memory size is 1024 MB.
4096
MB
4 MB
16384 MB
Go Back
Continue
Cancel

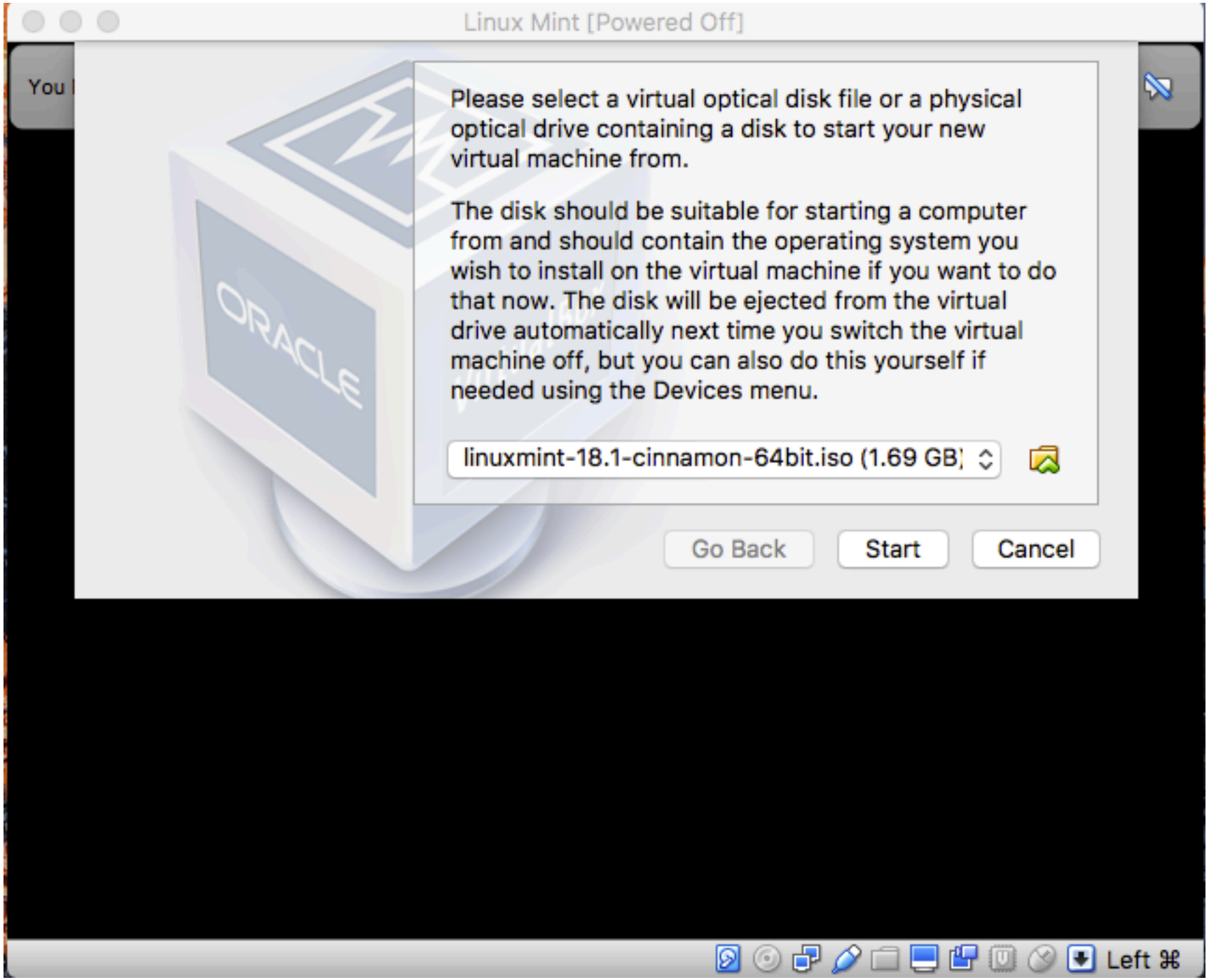

Oracle VM VirtualBox Manager
New
Settings
Discard
Start
Details
Snapshots
Hard disk
If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.
If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.
The recommended size of the hard disk is 8.00 GB.
Do not add a virtual hard disk
Create a virtual hard disk now
Use an existing virtual hard disk file
Empty
Go Back
Create
Cancel


Oracle VM VirtualBox Manager
New
Settings
Discard
Start
Details
Snapshots
Hard disk file type
Please choose the type of file that you would like to use for the new virtual hard disk. If you do not need to use it with other virtualization software you can leave this setting unchanged.
VDI (VirtualBox Disk Image)
VHD (Virtual Hard Disk)
VMDK (Virtual Machine Disk)
Expert Mode
Go Back
Continue
Cancel


Oracle VM VirtualBox Manager
New
Settings
Discard
Start
Details
Snapshots
Storage on physical hard disk
Please choose whether the new virtual hard disk file should grow as it is used (dynamically allocated) or if it should be created at its maximum size (fixed size).
A dynamically allocated hard disk file will only use space on your physical hard disk as it fills up (up to a maximum fixed size), although it will not shrink again automatically when space on it is freed.
A fixed size hard disk file may take longer to create on some systems but is often faster to use.
Dynamically allocated
Fixed size
Go Back
Continue
Cancel


Oracle VM VirtualBox Manager
New
Settings
Discard
Start
Details
Snapshots
File location and size
Please type the name of the new virtual hard disk file into the box below or click on the folder icon to select a different folder to create the file in.
Linux Mint
Select the size of the virtual hard disk in megabytes. This size is the limit on the amount of file data that a virtual machine will be able to store on the hard disk.
30.00GB
4.00 MB
2.00 TB
Go Back
Create
Cancel


Oracle VM VirtualBox Manager
New
Settings
Discard
Start
Details
Snapshots
Linux Mint
Powered Off
General
Name: Linux Mint
Operating System: Ubuntu (64-bit)
System
Base Memory: 4096 MB
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, KVM Paravirtualization
Preview
Linux Mint
Display
Video Memory: 16 MB
Remote Desktop Server: Disabled
Video Capture: Disabled
Storage
Controller: IDE
IDE Secondary Master: [Optical Drive] Empty
Controller: SATA
SATA Port 0: Linux Mint.vdi (Normal, 30.00 GB)
Audio
Host Driver: CoreAudio


Linux Mint [Powered Off]
Please select a virtual optical disk file or a physical optical drive containing a disk to start your new virtual machine from.
The disk should be suitable for starting a computer from and should contain the operating system you wish to install on the virtual machine if you want to do that now. The disk will be ejected from the virtual drive automatically next time you switch the virtual machine off, but you can also do this yourself if needed using the Devices menu.
linuxmint-18.1-cinnamon-64bit.iso (1.69 GB)
Go Back
Start
Cancel
Left ⌘


Linux Mint [Running]
Computer
Home
Install Linux Mint
Running in software rendering mode
Cinnamon is currently running without video hardware acceleration and, as a result, you may observe much higher than normal CPU usage.
There could be a problem with your drivers or some other issue. For the best experience, it is recommended that you only use this mode for troubleshooting purposes.
Menu
15:16
Left ⌘

Installing Linux

- On-screen instructions are easy to follow
- If you make a mistake, don’t worry!
  - Nice advantage of using virtual machine

Installing Grails

- Many ways to do this
- The following method works nicely for Linux and MacOS
- Requirements:
  - Installed JDK

- Software we will download
  - IntelliJ Ultimate
  - SDKMAN
  - Grails

SDKMAN! (The Software Development Kit Manager)

This tool makes installing Grails on any Bash platform (Mac OSX, Linux, Cygwin, Solaris or FreeBSD) very easy.

Simply open a new terminal and enter:

```
$ curl -s get.sdkman.io | bash
```

Follow the instructions on-screen to complete installation.

Open a new terminal or type the command:

```
$ source "$HOME/.sdkman/bin/sdkman-init.sh"
```

Then install the latest stable Grails:

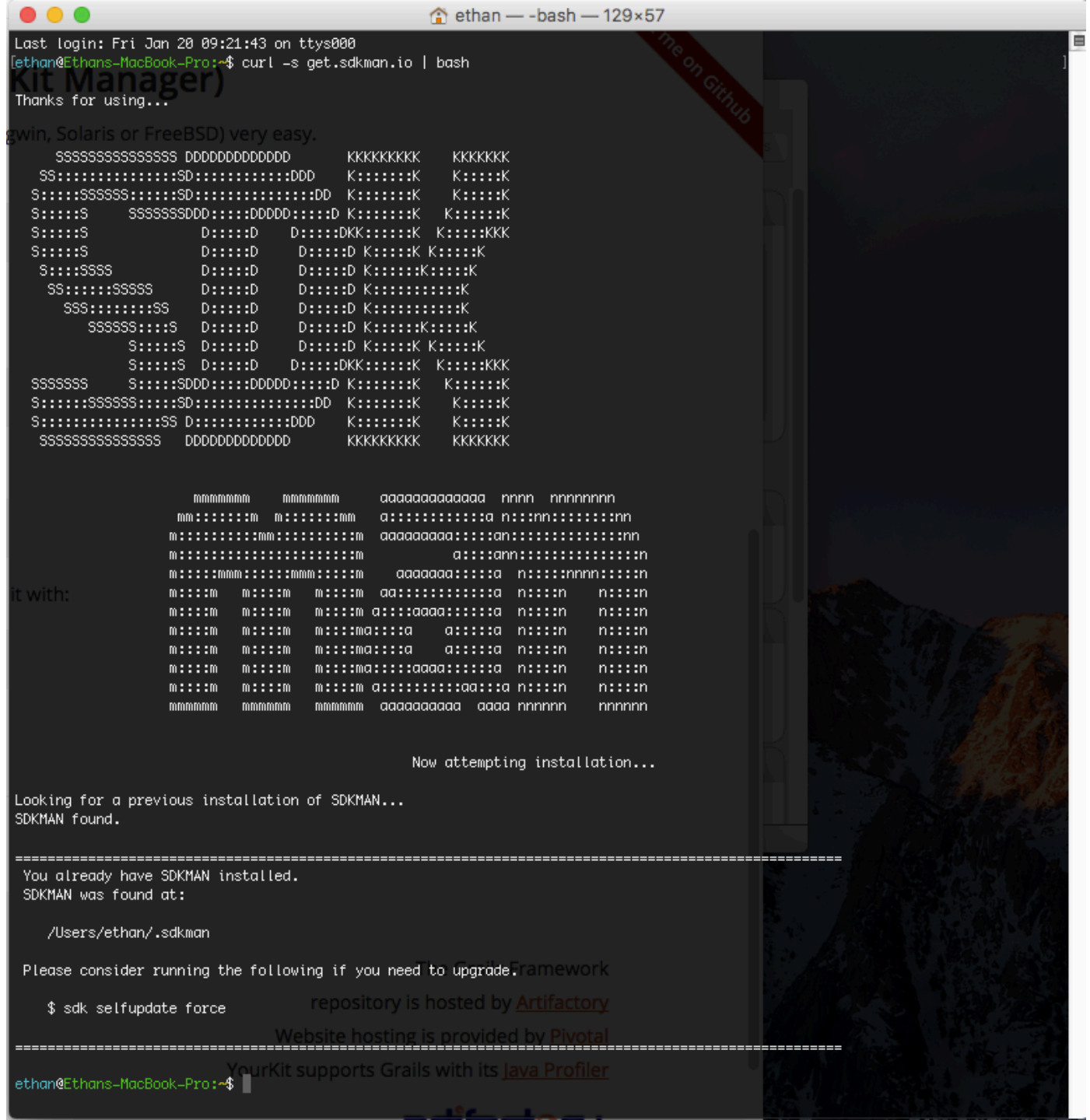
```
$ sdk install grails
```

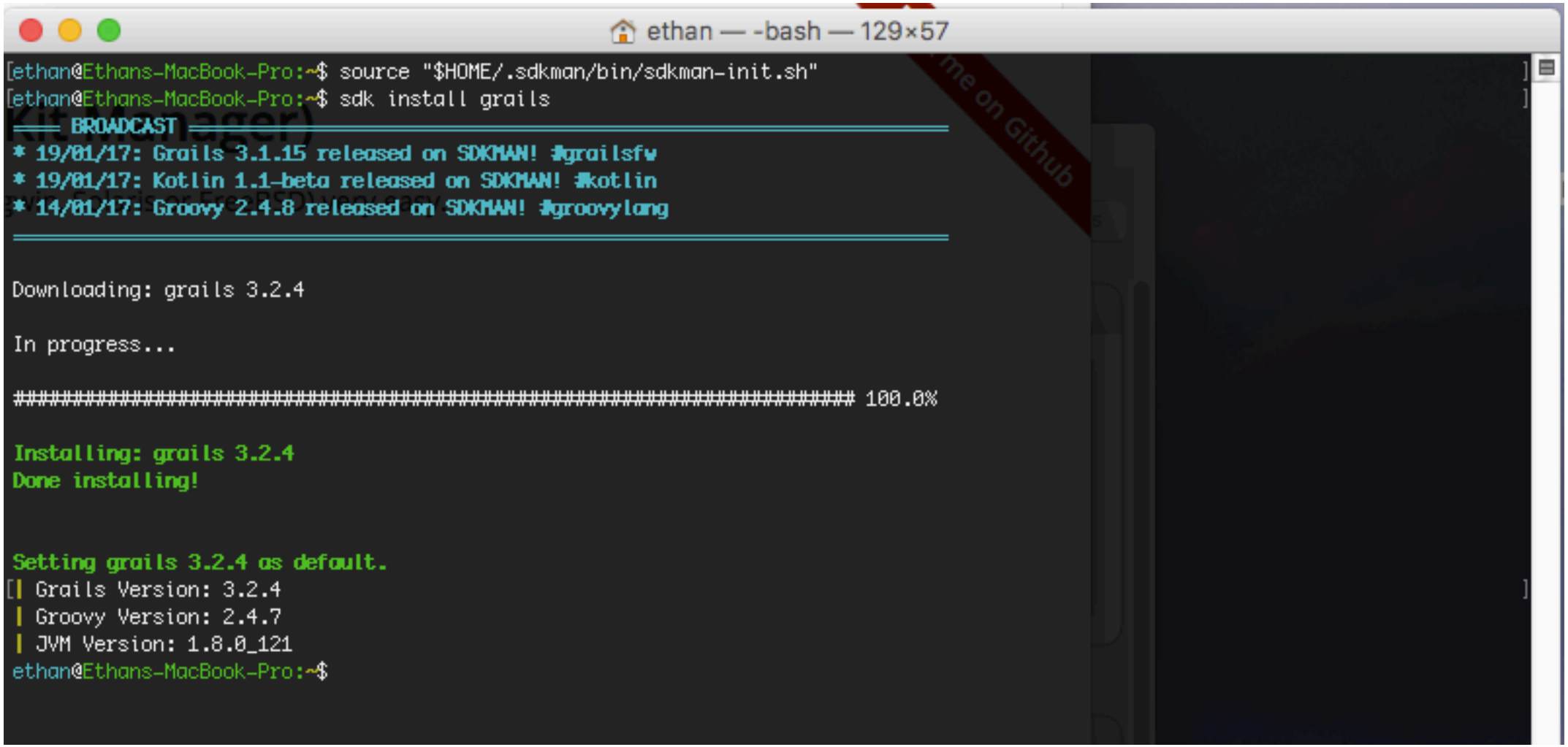
After installation is complete and you've made it your default version, test it with:

```
$ grails -version
```

That's all there is to it!

https://grails.org/download.html


ethan — -bash — 129×57
Last login: Fri Jan 20 09:21:43 on ttys000
ethan@Ethans-MacBook-Pro:~$ curl -s get.sdkman.io | bash
Thanks for using...
Now attempting installation...
Looking for a previous installation of SDKMAN...
SDKMAN found.
You already have SDKMAN installed.
SDKMAN was found at:
/Users/ethan/.sdkman
Please consider running the following if you need to upgrade.
$ sdk selfupdate force
ethan@Ethans-MacBook-Pro:~$

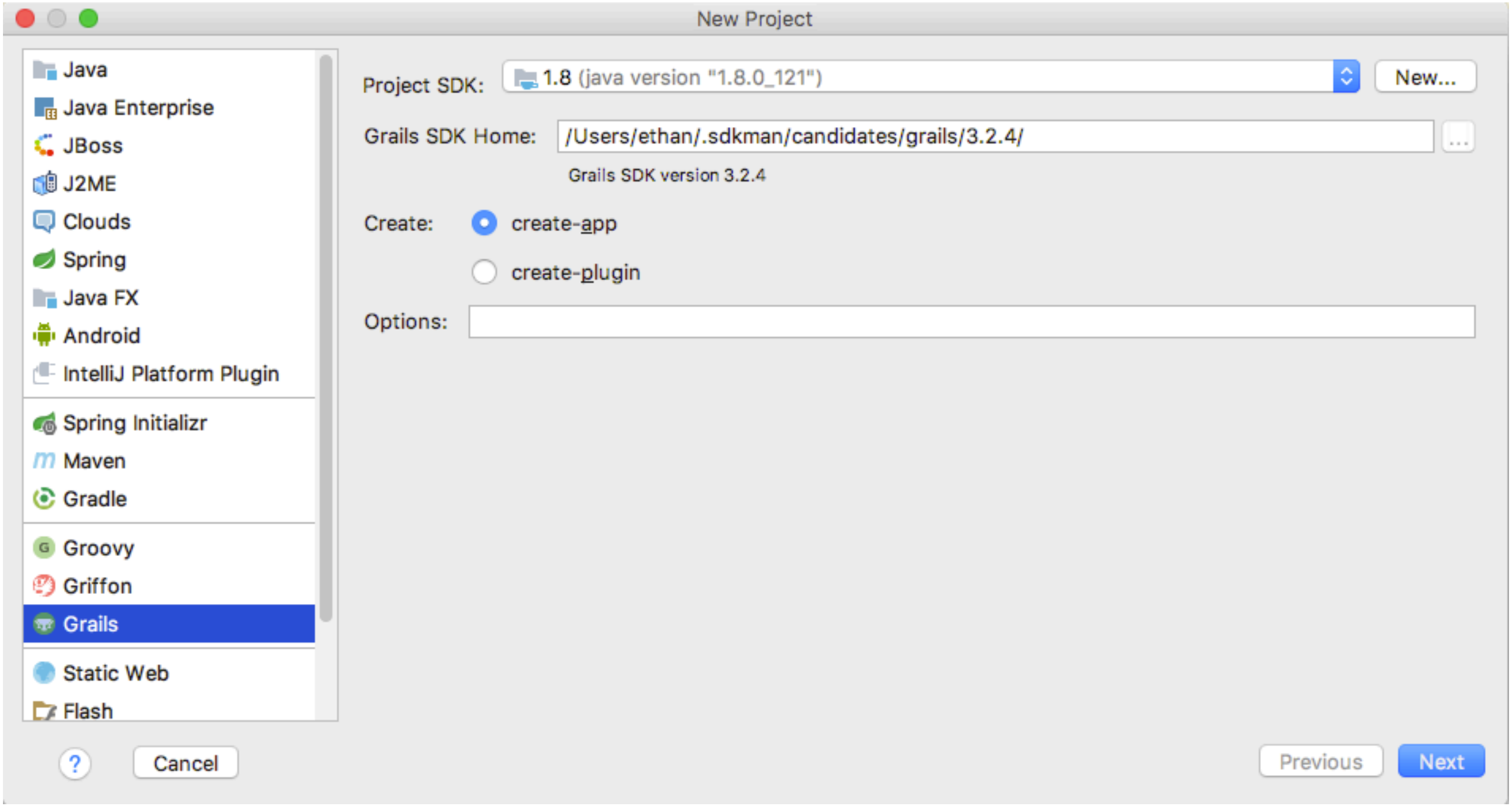

ethan — -bash — 129×57
[ethan@Ethans-MacBook-Pro:~$ source "$HOME/.sdkman/bin/sdkman-init.sh"
[ethan@Ethans-MacBook-Pro:~$ sdk install grails
BROADCAST
* 19/01/17: Grails 3.1.15 released on SDKMAN! #grailsfw
* 19/01/17: Kotlin 1.1-beta released on SDKMAN! #kotlin
* 14/01/17: Groovy 2.4.8 released on SDKMAN! #groovylang
Downloading: grails 3.2.4
In progress...
100.0%
Installing: grails 3.2.4
Done installing!
Setting grails 3.2.4 as default.
Grails Version: 3.2.4
Groovy Version: 2.4.7
JVM Version: 1.8.0_121
ethan@Ethans-MacBook-Pro:~$

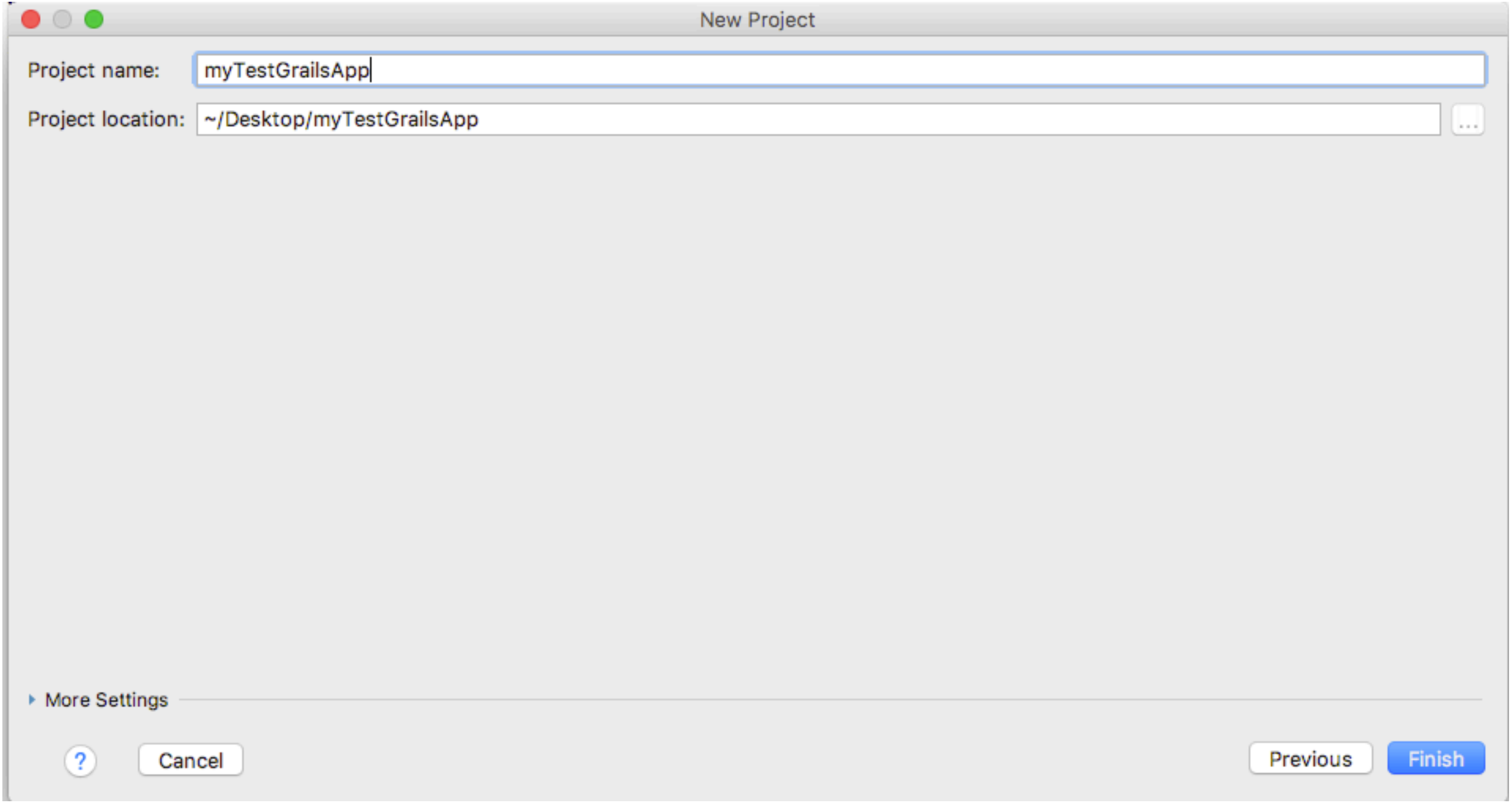
IDE Integration

- Grails documentation recommends IntelliJ IDE
- This is also my recommendation
- You can get the Ultimate Edition for free if you register as a student (highly recommended)

Creating an Application

- Applications can be created using Terminal or an IDE

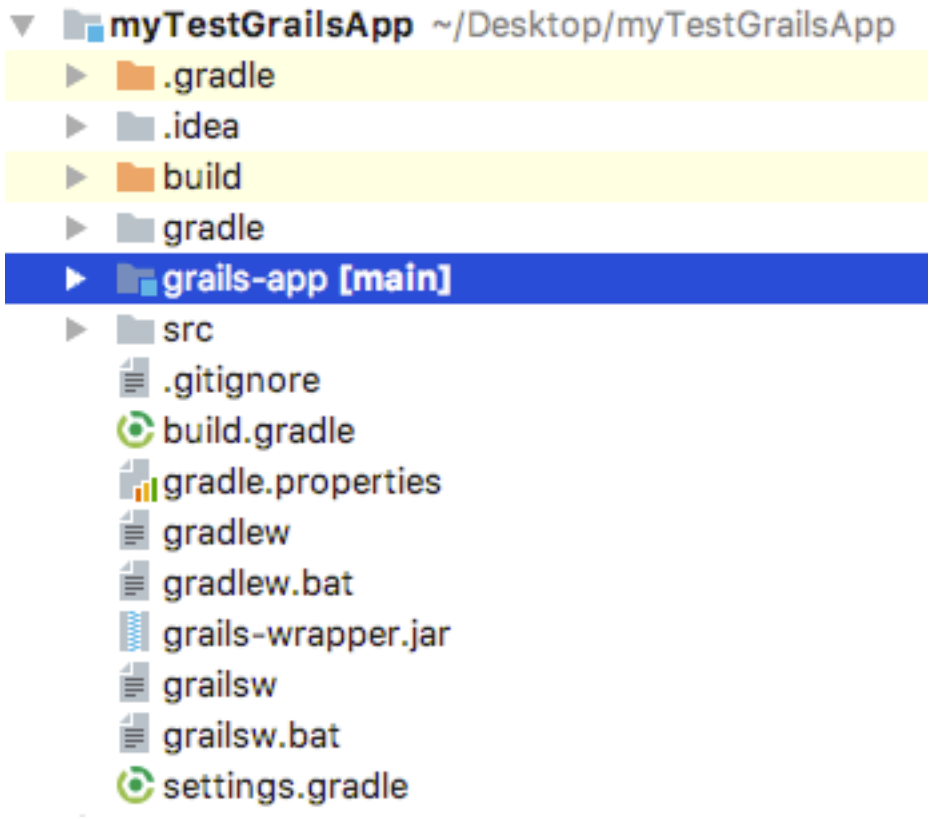

New Project
Java
Java Enterprise
JBoss
J2ME
Clouds
Spring
Java FX
Android
IntelliJ Platform Plugin
Spring Initializr
Maven
Gradle
Groovy
Griffon
Grails
Static Web
Flash
Project SDK:
1.8 (java version "1.8.0_121")
New...
Grails SDK Home:
/Users/ethan/.sdkman/candidates/grails/3.2.4/
Grails SDK version 3.2.4
Create:
create-app
create-plugin
Options:
?
Cancel
Previous
Next


New Project
Project name:
myTestGrailsApp
Project location:
~/Desktop/myTestGrailsApp
...
More Settings
?
Cancel
Previous
Finish

Grails Project Structure

Project Structure is automatically built.

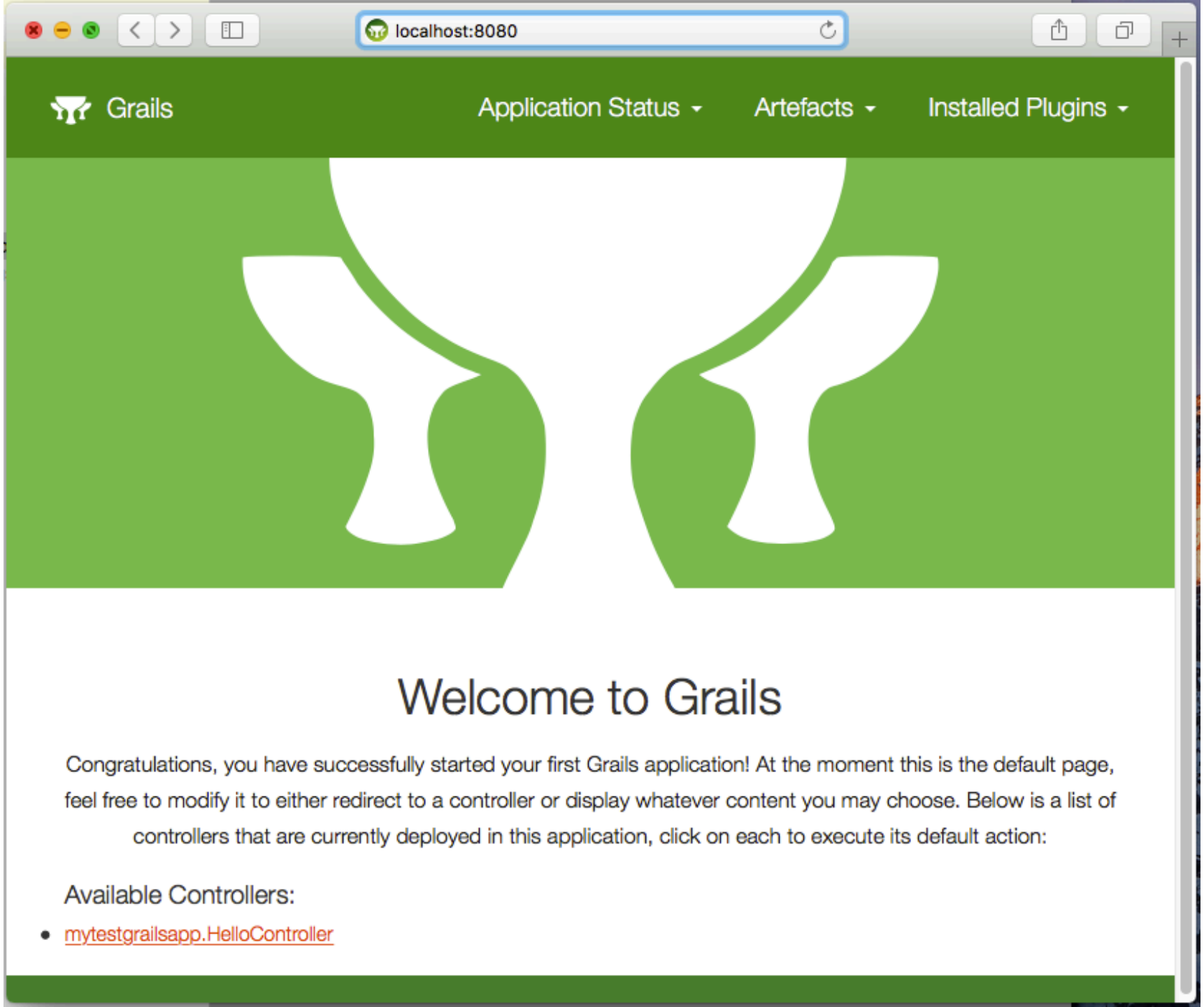
Build automation (Gradle) is automatically configured.



Running Application

- Run application from Terminal
- Navigate to project root directory
- Start grails by typing `grails`
- Start application by typing `run-app`


myTestGrailsApp — java ◂ java -XX:+TieredCompilation -XX:TieredStopAtLevel...
Last login: Fri Jan 20 09:22:59 on ttys000
ethan@Ethans-MacBook-Pro:~$ cd Desktop/
ethan@Ethans-MacBook-Pro:~/Desktop$ cd myTestGrailsApp/
| Enter a command name to run. Use TAB for completion:
grails> run-app
| Running application...
objc[29267]: Class JavaLaunchHelper is implemented in both /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/bin/java (0x100b634c0) and /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/jre/lib/libinstrument.dylib (0x100bf64e0). One of the two will be used. Which one is undefined.
grails> Grails application running at http://localhost:8080 in environment: development


localhost:8080
Grails
Application Status
Artefacts
Installed Plugins
Welcome to Grails
Congratulations, you have successfully started your first Grails application! At the moment this is the default page, feel free to modify it to either redirect to a controller or display whatever content you may choose. Below is a list of controllers that are currently deployed in this application, click on each to execute its default action:
Available Controllers:
mytestgrailsapp.HelloController

Coming soon…

- Selected Grails tutorials
- Example code on the course GitHub
- Some basic front-end implementation
- Integration with Java classes (important)