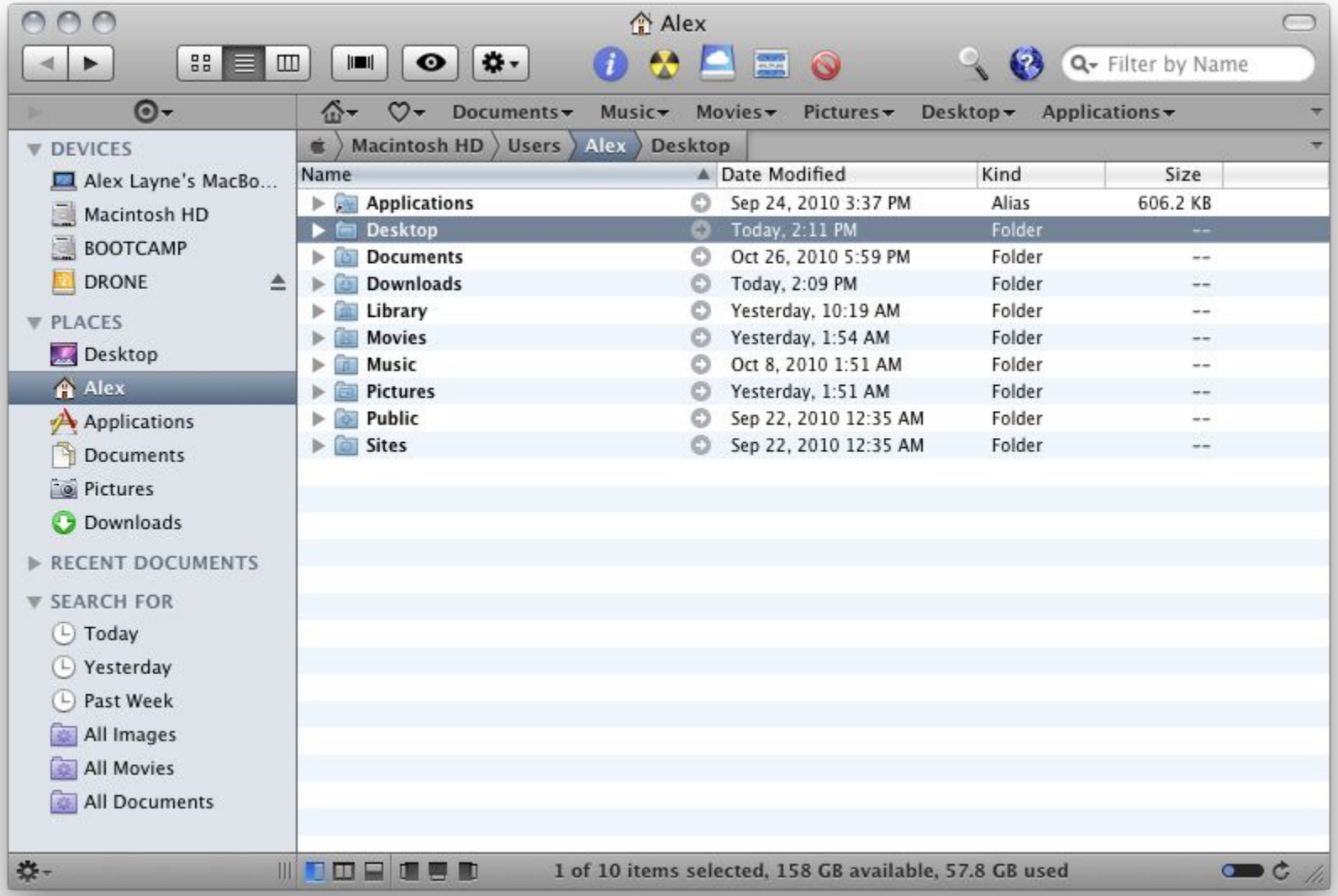
Unix & Bash Basics

Stat 133 by Gaston Sanchez

GUI & CLI

Windows and Mac use a Graphical User Interface (GUI) for you to interact with the Operating System



MAC Finder: GUI to intereact with the OS

About GUIs

Easy to learn

Rely on visual displays

Can be extremely useful

Have improved the friendliness and usability of computers

GUIs are excellent for ...

Photo editing

Document Layout

Browsing the web

Graphic design

Watching movies

Playing Games

GUIs Trade-offs

They are limiting

They don’t allow you to have more control over what your computer can do

Some operations are labor intensive and repetitive

You use clicking & dragging with the mouse, which reduces reproducibility

GUIs disadvantages

Lack of repeatability

Lack of reproducibility

Some tasks may be labor intensive

Limit analyses on a cluster of computers

Instead of a GUI,
you can use a
Command-Line Interface
(CLI)

Don’t touch that mouse (or track pad)!

CLIs

CLI: Command Line Interface

Instead of using a GUI we can use a CLI

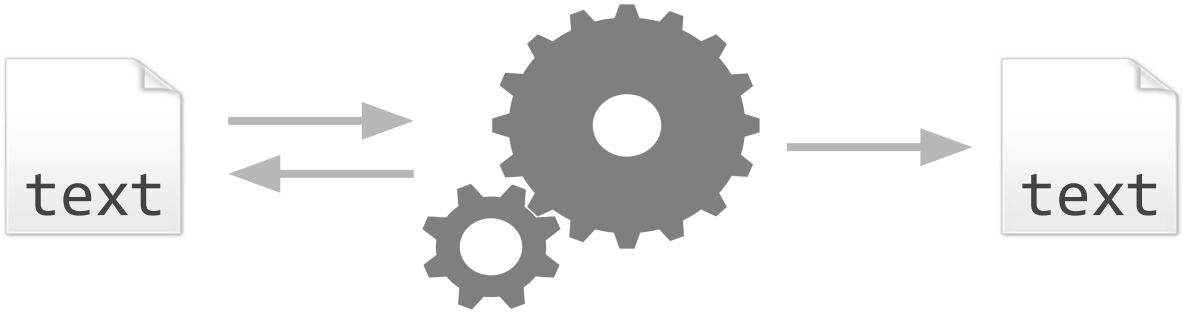
By typing commands we perform tasks on the computer

CLI

The command-line is a text-based interface

The user issues commands in the form of text

One of the earliest ways of interacting with a computer



Why use the Command-Line instead of a GUI?

Software development & System Administration

Don’t have or don’t need a GUI

Servers, utilities, and other programs run on a server somewhere without a monitor

The CLI started in 1971 with the Thompson shell for Unix

CLI -vs- GUI

More convenience
Less Control

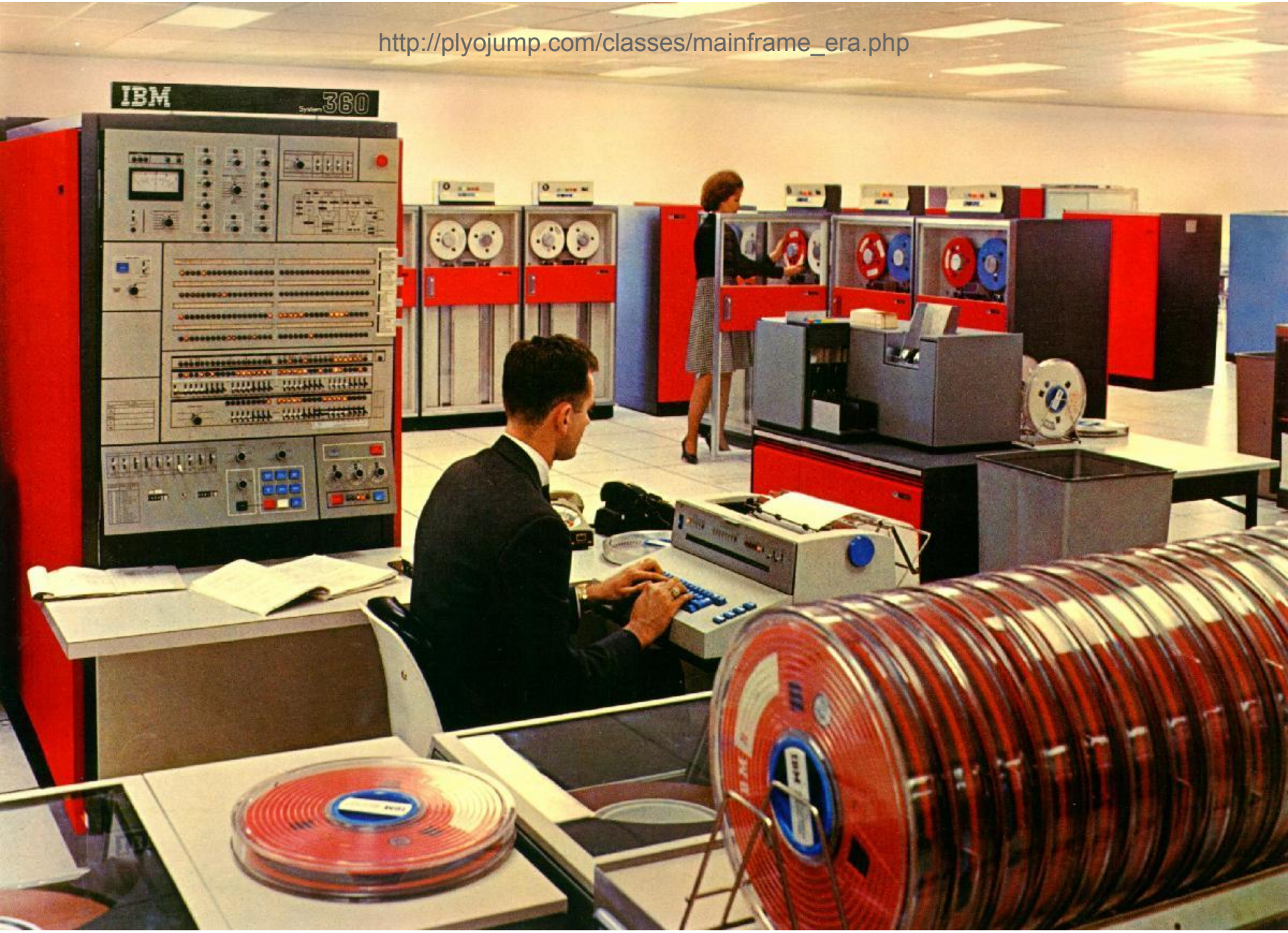
GUI is like automatic transmission

Less convenience
More Control

CLI is like manual transmission

Unix-like & Bash Command line

To talk about the CLI we need to talk about UNIX


http://plyojump.com/classes/mainframe_era.php
IBM
System 360

What is UNIX?

Unix is an Operating System

Developed by AT&T employees at Bell Labs (1969-1971)

Predecesor MULTICS: Multiplexed Information and Computing Service

MULTICS was designed to manage remote login process

MULTICS became a huge and complex project

And Bell Labs decided to cancel it

http://plyojump.com/classes/mainframe_era.php

What is UNIX?

Employees who worked on MULTICS kept secretely developing a small and simpler version system

Originally, their version only supported one single user

The new project was called UNICS: Uniplexed Information and Computing Service

UNICS became quickly able to support multiple users

It was then renamed to UNIX

What is UNIX?

Due to an antitrust case, AT&T was forbiden to enter the software bussiness

AT&T starts give UNIX away in 1975 with the first public version “System Five”

Since then, many branches and releases have kept the development of UNIX alive

The most famous one is probably the Berkeley Software Distribution (BSD) that came out in 1977

Some Versions of UNIX

Linux (open source)

Sun Oracle “Solaris”

IBM “AIX”

Hewlett-Packard “HP UX”

Apple’s “Mac OS X”

Etc

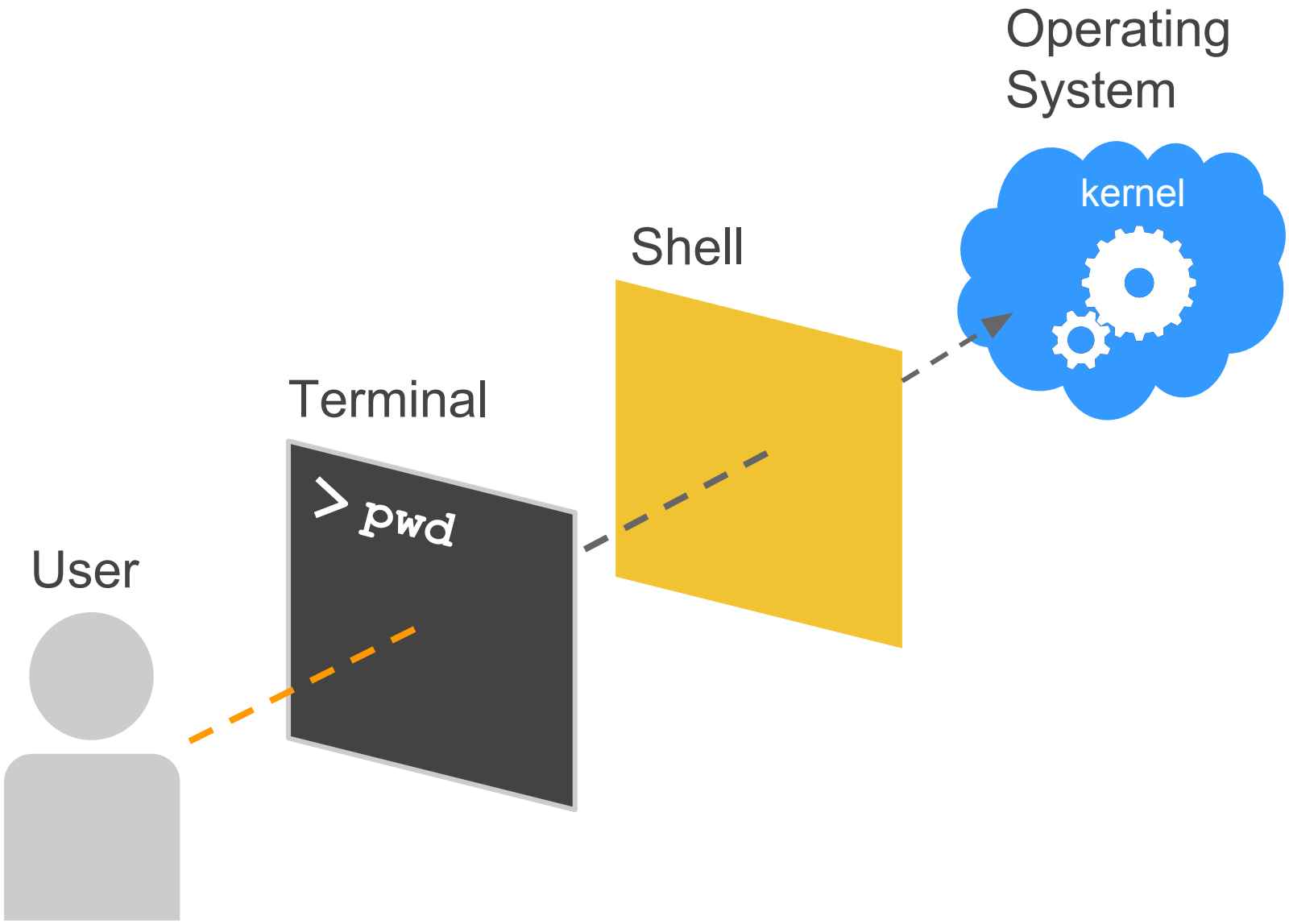
If you hear UNIX now it can mean all of these things

Kernel

Shell

CLI

Terminal


Operating System
kernel
Shell
Terminal
> pwd
User

About the Kernel

The **Kernel** is the **core** of the operating system in Unix

It takes care of allocating time and memory to programs

It does very fundamental root level management

About the Shell

The **Shell** is the outer layer of the OS

That's what we see when we open up a terminal window

We are working in the Shell

It interacts with the user and sends requests to the kernel

The Kernel sends results back to the Shell

Command-Line Interpreter (Interface)

The command line is a text-based interface where you type commands and direct text-based input and output to the screen, to files, or to other programs

The environment you use is called a shell or a command-line interpreter

The main programs we're going to be concerned with are the **Terminal** and the **Shell**

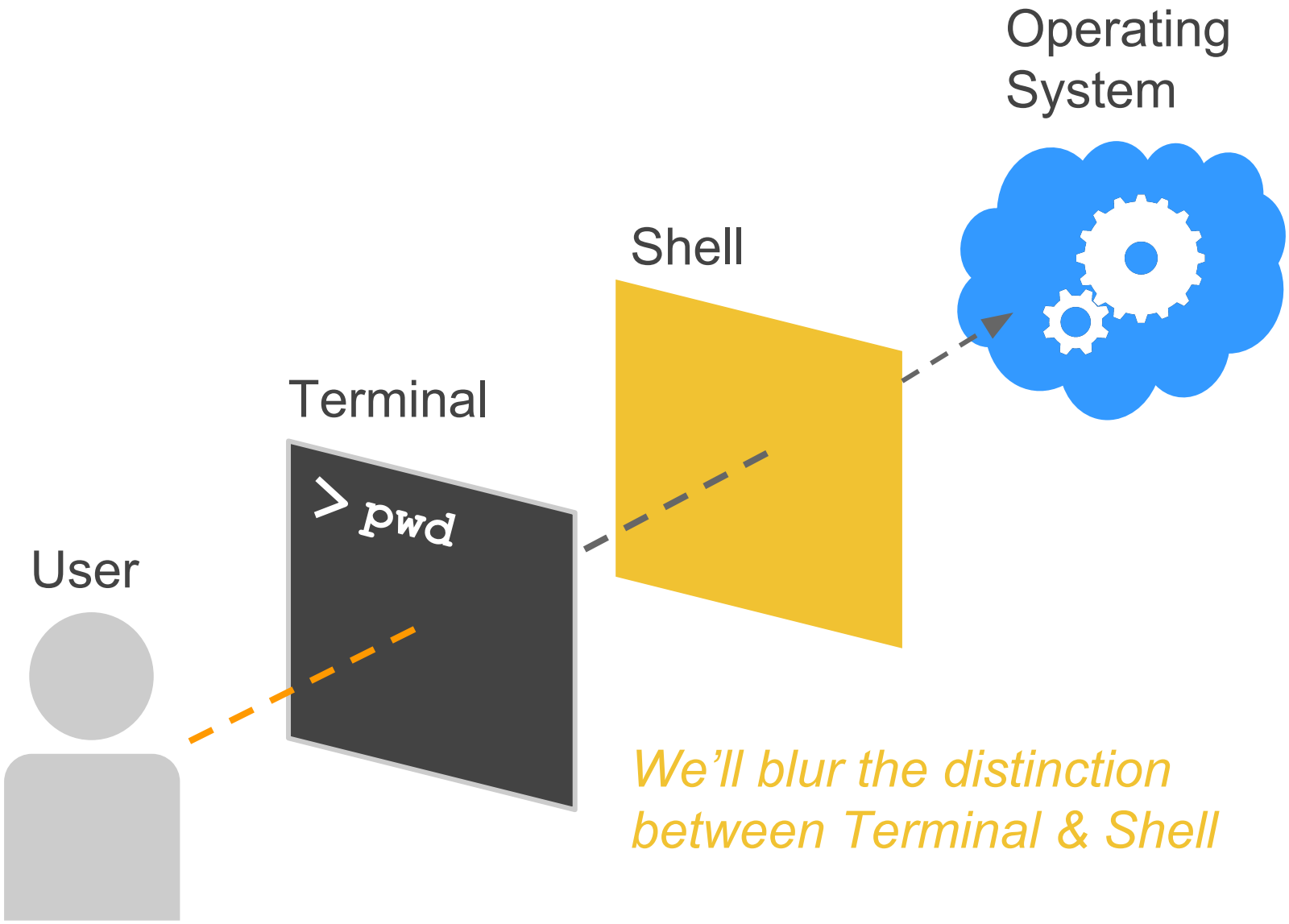
Shell

Shell: program that takes commands from the keyboard and pass them to the Operating System to be executed

- bash
- zsh
- ksh

Terminal

Terminal: (a.k.a. terminal emulator) program that opens a window and lets you interact with the shell



We'll blur the distinction between Terminal & Shell

Terminal


>_
THE TERMINAL

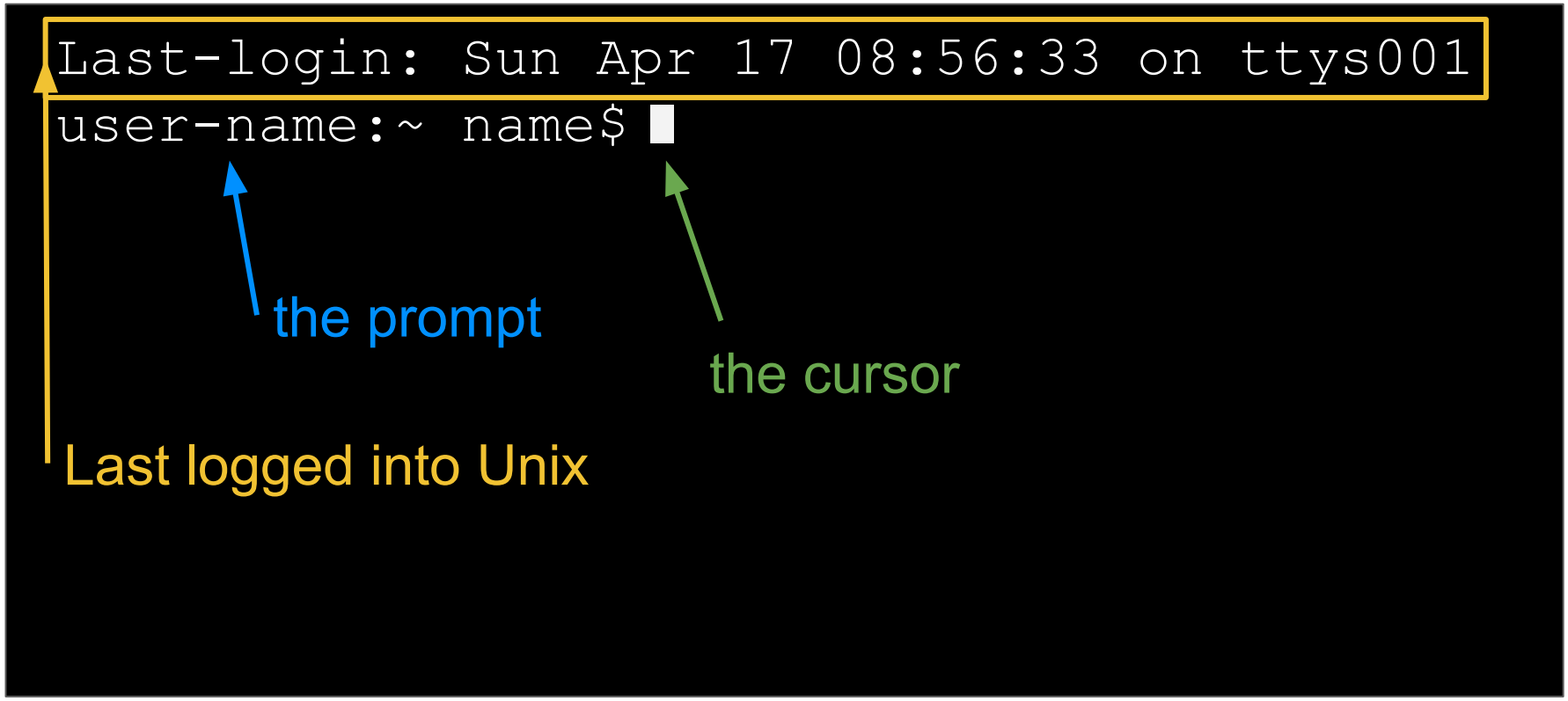
The terminal

The way to interact with UNIX is from the command line

If you use Mac, access to the command line is with the **Terminal** Application

That's why many people use the terms "command line" and "terminal" as synonyms (although they're not the same thing)

The Terminal



Some shortcuts

Up / Down arrows: review previous commands

Ctrl + A: move cursor to start of line

Ctrl + E: move cursor to end of line

Option + click line: move cursor to click point

Ctrl + L: clear screen

Tab: try to complete the command or file

Terminal

Where is it?

How to open it?

Mac:

Applications > Utilities > Terminal

Windows:

Use Git-bash

About the shell

The shell does 4 simple things:

- displays a prompt in the terminal window (waiting for commands)
- reads your command
- runs the command
- prints the output, if any, to the Terminal window

BASH

The most common type of Shell is **Bash**

BASH: Bourne Again Shell

Default shell for Linux

Bash is usually the default shell on Mac

It is also used in Git-bash

About the terminal

The terminal's job is basically to:

- open windows and tabs
- manage shells
- resize windows
- change colors in windows
- handle copy-paste operations

Command Line Interface

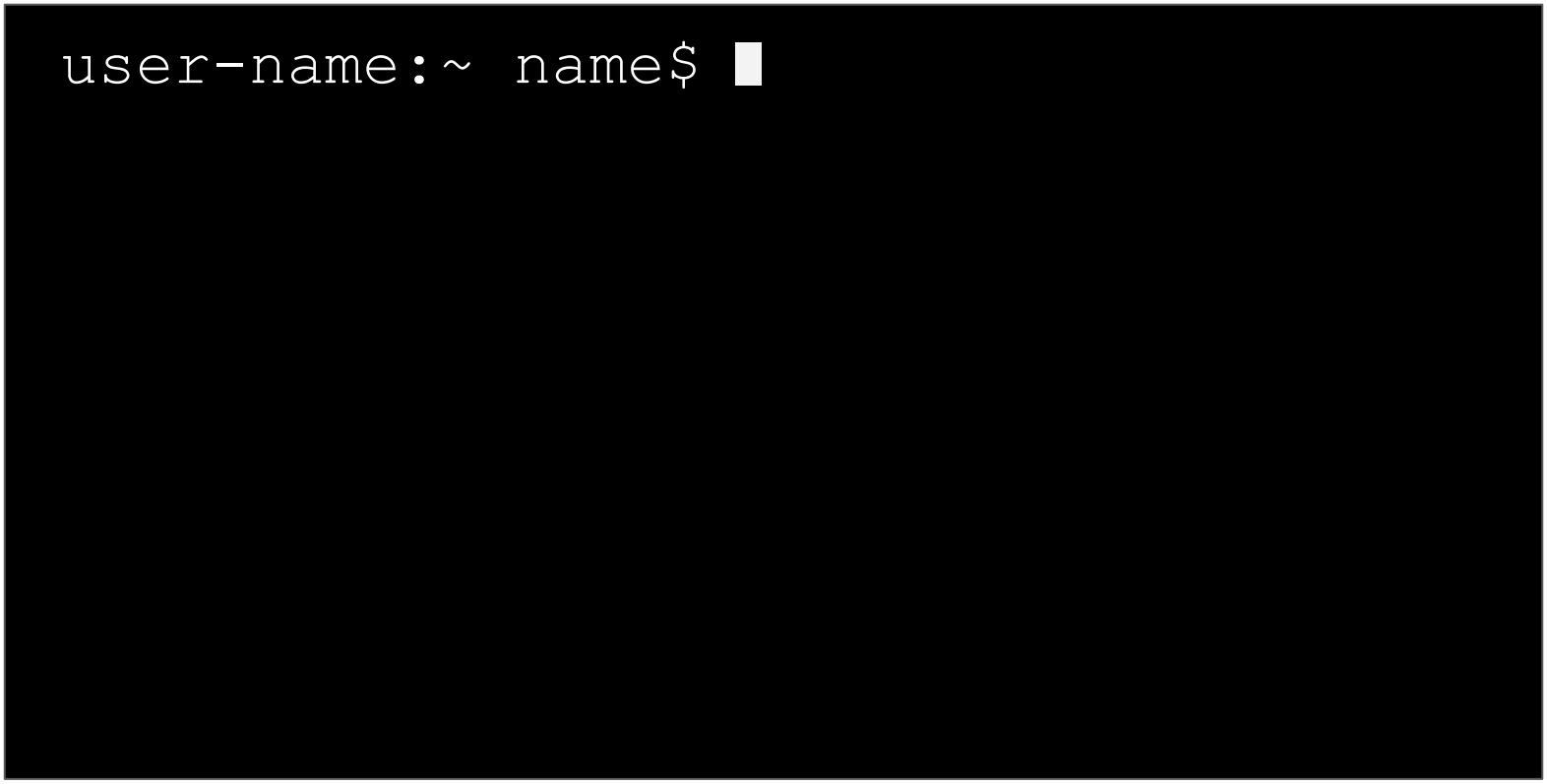
It is text-based

You type commands

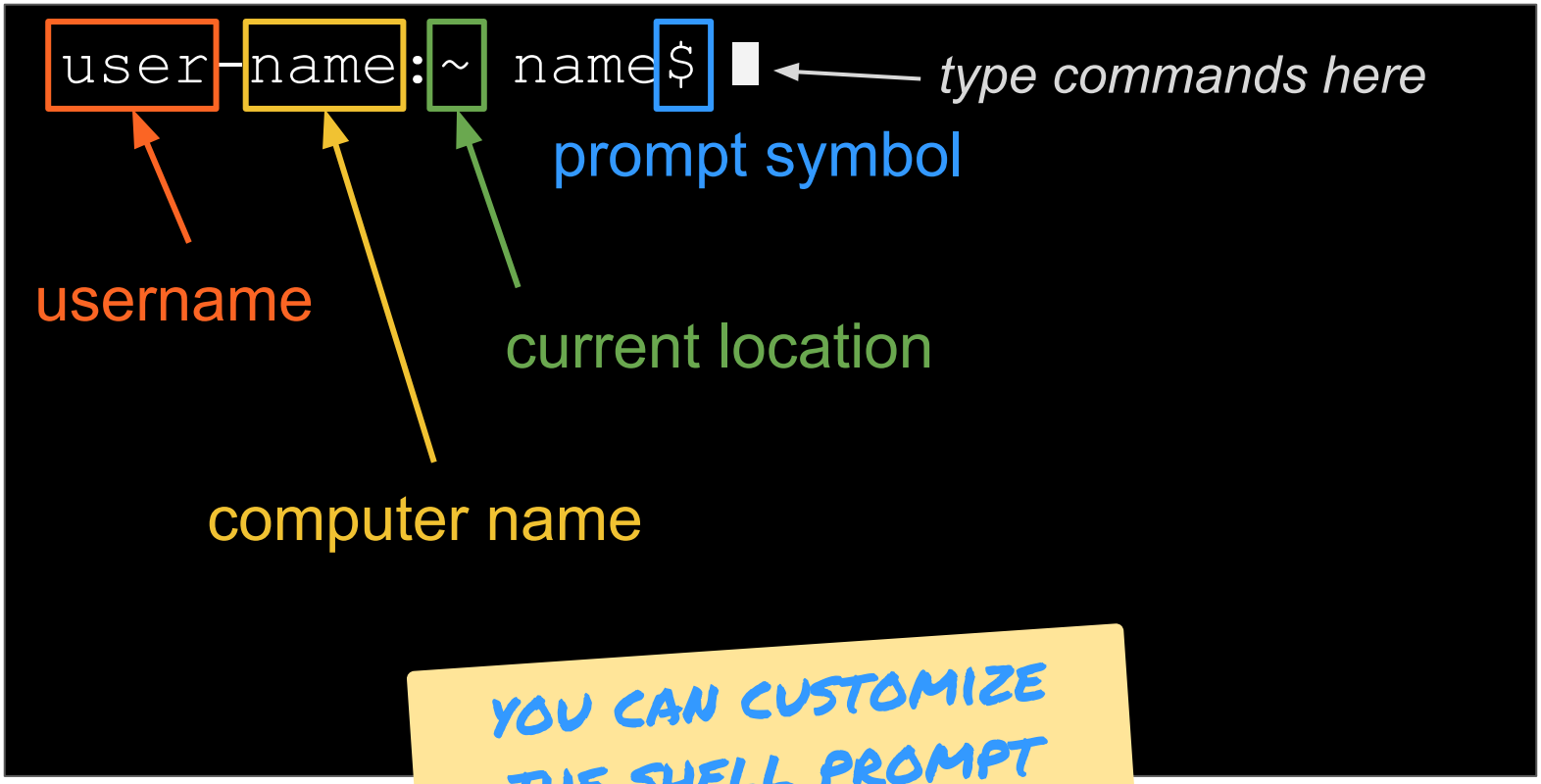
Commands are executed

… keep typing commands

The Prompt



The Prompt


user-name:~ name$
type commands here
prompt symbol
username
current location
computer name
YOU CAN CUSTOMIZE THE SHELL PROMPT

Commands

Interacting with the shell requires using commands

General Command Syntax

```
ls -lh /usr/bin

sort -u users.txt

grep -i "needle" haystack
```

```
command option(s) argument(s)
```

General Command Syntax

```
ls -lh
```

```
command option(s) argument(s)
```

General Command Syntax

```
ls /usr/bin
```

```
sort users.txt
```

```
command option(s) argument(s)
```

General Command Syntax

```
ls

sort

grep
```

The command is the program you're running

```
command option(s) argument(s)
```

General Command Syntax

```
-lh

-u

-i "needle"
```

Options tell the program how to operate

```
command option(s) argument(s)
```

Command Options

Options tell a command how to operate

Start with a dash (some with double dash)

Usually one letter

More than one offered by most commands

Options

```
ls -l -a -h /var/log
```

```
command option(s) argument(s)
```

Options

```
ls -lah /var/log
```

You can combine various options

```
command option(s) argument(s)
```

General Command Syntax

```
/usr/bin

users.txt

haystack
```

Arguments tell the command what to operate on

```
command option(s) argument(s)
```

Command Arguments

Arguments tell the command what to operate on

Usually:

- File or directory
- Set of files or folders
- Path

Command Structure

```
command␣options␣arguments
```

Single space *Single space*

ALWAYS IN THIS ORDER!

Command Structure

```
command    options    arguments
```

Name of the command

Always a single word

The thing you want to do

Command Structure

```
command    options    arguments
```

Options are optional

Not all commands require options

Controls the behavior of the command

Specified with single or double dashes

Command Structure

```
command    options    arguments
```

What to operate on:

- File(s) or directory(ies)
- Path

Some examples (with and w/o options)

```
echo 'Hello World'

echo -n 'hello world'

ruby -v

ruby --version

ls -l -a -h Desktop

ls -lah Desktop
```