2.1 State and 2.2 Modularity and Applications

Notebook:	How Computers Work [CM1030]		
Created:	2019-10-09 10:09 AM	**Updated:**	2019-10-13 3:45 PM
Author:	SUKHJIT MANN		
Tags:	Bug, library, Locking, Metadata, Modularity, State		

Cornell Notes	**Topic:** 2.1 State 2.2 Modularity and applications	Course: BSc Computer Science Class: How Computer Work [CM1030} Date: October 13, 2019

Essential Question:

- What are states, how they present in various aspects of the computing experience and to that how software modular affects our understanding and debugging of such a system?

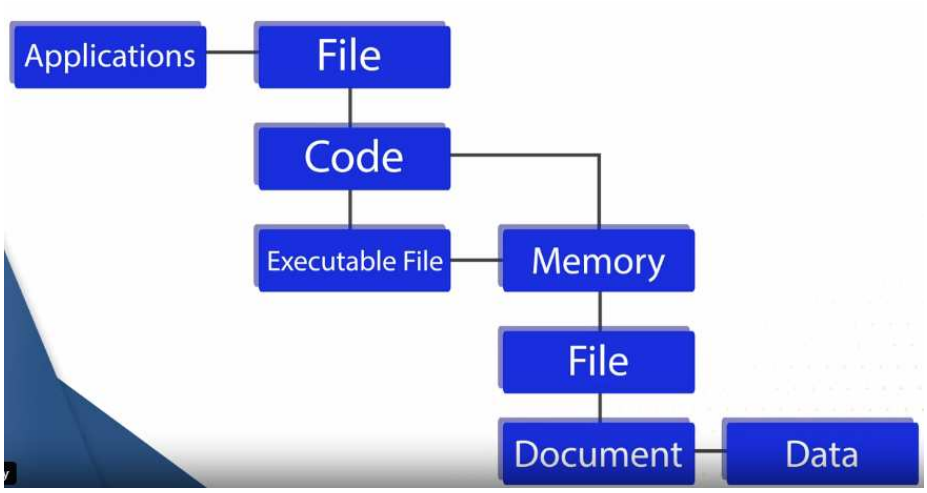
Questions/Cues:

- What is a state?
- What are two states of a computer?
- How does a computer change between states?
- What are basic states of a video player?
- What are the drawbacks of having internet connected applications?
- What is a website where you go to buy something?
- What are states of a e-commerce site?
- What does a state depend on?
- What does numbers and states have in common?
- What is a file?
- What are files on a hard disk?
- What can files contain?
- What is metadata?
- What is a permission system on a Operating System?
- What is file locking?
- What is Modularity?
- What are drivers?
- What are some tasks that the Operating System handles?
- What are libraries?
- What is a bug and what is debugging?
- What are some simple steps to debugging?

Notes

- State = temp config of a Comp system, affects how it responds to interaction
- Basic Comp States = on or off

- Any given time, comp in one of several states, changes between b/c of user input or possibly through own internal processes
- Video player = play state, pause state, stop state. For online video, sometimes a buffer state.
- Internet connected apps heavily relies on the strength of internet connection, if it fails, apps interrupted, freeze, crash or produces unexpected behavior ie. email doesn't know what to do with a half email so crashes/ refuses to fetch emails properly and update client.
- Buying website = e-commerce (electronic commerce) website
- Home page → Search page (# of results, possible products) → looking at particular product→ Add to/Added to Shop cart → Possible Login page → Logged in and item added to shop cart → Checkout state → Delivery address→Card details→Delivery method→Click purchase now→Processing Card transaction→Payment Sucessful→ Item bought
- State depends on how Comp Memory is set up
- Comp Memory stores billions of #'s, changing any #'s = changing state of Comp
- # of comp states comp can have is astronomical, more than # of atoms in the universe
- Progs written to avoid nonsense states
- Modern apps connect to internet to get data or checksum info, internet breaks or not logged in, stuck in frozen loop
- Start of prog = default state, cleared of last state, best fix for strange state; **turn it off and turn it on again**
- File = abstraction of some data, doc, pic or song
- Files on a hard disk = loads of patterns on disk surfaces
- OS presents the patterns of raw data as set of files in a folders
- Files = Data, words, pixels, or audio samples and associated metadata
- Metadata = data of file, when created or who owns file
- Metadata not part of file, stored by OS
- Metadata used by OS when dealing with file itself, recent files, OS looks at Mod dates
- Permission System = prevents opening of other ppl files
- File lock = locking files when apps uses them so other apps cannot open them, app opens file sets the locked metadata and when app closes file it unlocks it
- File in Close state on HDD→App opens file and its locked→App copies file to memory to edit(while keeping HDD copy open and locked until copying is complete)→ File on HDD closed and unlocked and app now shifted to work on file data in memory→ Closed, unlocked, unchanged file on HDD and copy on memory that is changed→ App opens HDD file again and locks it to save changes to file→Changes copied back to HDD and HDD file is closed and unlocked again→App closes file and back to closed state of original file, contents changed
- If app crashes between editing and saving to file, data lost, but modern systems use backups to prevent this
- Modularity = splitting into smaller chunks called modules, about creating systems of simple modules that interact with each other
- Drivers = software that controls hardware, independent of any application
- Code is stored as bit in memory, files before app starts just like doc, instructions to CPU(code) just another type of data



- OS tasks = drivers, writing to files, interacting with networks, and drawing to screen.
- Libraries = bits of code, work with app to make app can use surface, sometimes part of exe file as app. Can be separate file called dynamic library that interacts with main exe file
- User interface library = code to display buttons, menus, and other interface items
- Resource file = data applies to app as whole, not to individual docs, ie. to make UI look right or store setting preferences
- Bug = error in Comp prog
- Debugging = getting rid of bugs, or errors
- Understand the System→ Look, don't Think (carefully looking at what Comp is doing and gathering info)→ Make it fail (Know when problem happens, test system and reproduce bug reliably)→ Only change one thing at time

Summary

In this week, we looked at computer states and in turn states of various applications we use on said computer. Alongside this, we discussed the importance and impact of modularity of a application system and its various components which work together to provide a user their experience. Lastly, we discussed the effect of these components when bugged and producing erratic results and method of debugging we would use to bring the system to a functional working state.