**OVERVIEW:**

* There are a lot of similarities and similar problems that a lot of developers are having.
* So many people app needs: authentication, maybe logger for error messages, connect to DB, web application…
* We want core features to our app, but all these common things anyway no matter what we doing, but we can get help.

Maybe open source project to solve some of these problems…yay! This is model BEHIND **LIBRARIES**

* Libraries used to solve common problems and tasks that these multiple app devs trying to solve.
* Goal to help common challenges so developers can write and solve main problem of their code.
* Reusable, and have well defined programming interface like certain classes and methods that need to be called to get things done.
  + Can just call out library and use api and get job done.
  + Let’s say you are carpenter to build table, have everything you need but need some help.
  + You want a too lset with hammer, screw driver, can pull out whatever tool you want as you build table. TOOLBOX is library there are tools you use when need them .
* Great for wide range of purposes, can use same “toolset” to build boat or table.

**FrameWork:**

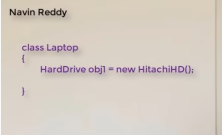
* have hundreds of people building tables. Each person when creating table go through same set of steps. Like first lay planks, saw wood, hammer nails,….
* Point is we have TOOLBOXES to help carpenters to handle common tasks
* Usage of tasks repeated over and over, begin to see COMMON PATTERNS.
* SO this pattern is for every table carpenter….
* HoW TO MAKE THIS EASIER?
  + Create table manufacturing facility like a machine that positions plank ready for you to cut. The machine than takes cut pieces with hammer that can hammer nails and stuff….
  + HAVE STREAMLINED PROCESS THAT ALL CARPENTERS HAVE TO DO ANYWAY
  + Provided overall framework for carpenters to operate.
  + Carpenters don’t have FULL control of what they building like library,
    - Told exactly how things are going to goal just follow steps.
* FRAMEWORKS provide patterns that make it easier to operate with confines to those patterns.
* In Java have ramework for building web applications.
  + Ex: this will listen to web requests, handles sessions, call your code to handle requests.
  + YOU CALL THE LIBRARY, BUT THE FRAMEWORK CALLS YOU.
  + Makes things simple, but loose flexibility.
  + Web app framework only to build web apps.

**WHAT IS SPRING FRAMEWORK**

* Tackle common application problems.
* Provides patterns and structure for Java applications and handle common things that most devs need to do to make java application
* THREE MAJOR THINGS
  + APPLICATION CONTEXT AND DEPENDENCY INJECTION
    - Spring manages your objects instances.
    - Can tell to create one instance of class for whole app
    - Manages instances as well, and connects them together
    - Every class declares dependencies that it needs
      * Ex: CLASS A need instance of class b and c.
  + Data Access
    - Standard way to use db in java utilize JDBC (V painful)
    - Most common things that developers need access too.
    - Spring helps wit connectivity, query, transaction management, etc.
    - Works with JDBC, but hides all the nasty stuff
    - Simplifies connecting and working with databases.
  + SPRING MVC (web framework)
    - Lets you easily create web apps and rest apis using same spring model

[**Spring Framework Tutorial | Full Course**](https://www.youtube.com/watch?v=If1Lw4pLLEo&t=213s)

**Dependency Injection**

* What and why?!!
  + What we do in sw is build objects and code… those objects dependent on some other objects.
    - Example: lets say want to build laptop. Laptop has certain parts like ram ,hard drive, screen. Apple screem from Samsung, hard drive from itachi, RAM from sanders. All these companies help one company build machine
    - If want to build project and an obj dependent on some other obj, we not building everything by ourselves.
      * In java we create class than object lets say laptop
      * In Class Laptop we need class hardrive lets say
        + We use new keyword to get access to it….
        + In this example we are hard coding by saying new hitachiHD[] this is **tight coupling** because specified, not very flexible can’t use another HardDrive.
        + We want something else to give us that dependency, the laptop object is dependent on the harddrive object that is a **dependency.**
        + Want to **inject** the HD obj inside this laptop class. WE CAN DO THIS bY USING SOME EXTERNAL SERVICE that will inject this dependency.
        + Possible by: **Dependency Injection Containers**

**These are responsible to create obj for you, and than injecting in your class.**

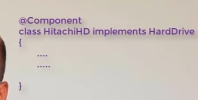
**Where to inject and how to create object?**

**That’s where we need CONFIGURATION**

* + - * + In Java have spring framework, and can configure everything usin XML.
        + We create xml file and mention hey if someone asking for hard drive,GIVE THIS OBJECT.

We can edit xml so not tight coupled.

* + - * + IN JAVA we don’t really want to use xml though, we can use spring boot.

**Can have some class as Hitachi or Samsung hard drive. At top of these classes we add @component which makes them dependent, so it makes it a component of a spring framework and will be generated as per requirement.**

**Now what about this class? How to mention that you want this object?**

Notice in autowired, we don’t have to say “new hitatchi…”

WE CAN on top of hard drive add @autowired.

What this meansi s that someone asking for hard drive and we have this component so we can auto connect.

THIS MAIN IDEA BEHIND DEPENDENCY INJECTION.

SO Why WE NEED THIS?

We want loose coupling not tight

Good for testing

We always test sw as whole, we should also test each component (unit) lets say want to test laptopcv

So use dependency injection implement to

Test better

Maintain better

[**What is spring boot?**](https://www.youtube.com/watch?v=35EQXmHKZYs)

* Still using spring framework, but Spring boot is middle man
* In between have Spring Boot to help you
* Boot say I give you dependency and config, give production ready app
* So don’t have to do any config.
* Open source java based framework used to create micro service.
* Used to build stand alone and production ready spring applications.
* MICROSERVICES?
  + Architecture allowing devs to develop and deploy services independently.
  + Each service running has own process, achieves lightweight model to support business applications.
* WHATS SPRING bOOT?
  + Provides good platform for java devs to dev stand-alone and production-grade spring apps that you can just run. Can start with min configurations without need for entire Spring config setup.
  + Helps avoid complex XML config
  + Develop production ready Sspring app in easier way/
  + Reduce dev time and run app independently.
  + Automatically configures app based on dependencies you have added to proj by using @EnableAutoConfiguration annotation.
* <https://www.youtube.com/watch?v=vtPkZShrvXQ>
* Platform to build back end / web applications
  + Has a lot of dependencies which provides flexibility
  + Don’t have to configure and setup, just have to install dependencies and start implementing whatever.