1. Vague and complicated things means no question of applying in project.
2. I know architecture
   1. We know layering.
   2. It means we can check each layer independently
      1. That is reason we learnt the syntax called commandline runner ..
3. We want to understand Spring Boot applications
   1. Spring boot is a cover on Spring
      1. Loose coupling
         1. Interface
            1. Until we know interfaces no question of loose coupling
            2. 2 is immune when 3 changes.. interfaces 1 is a standard.
         2. Inversion of control
            1. That is we can ask spring framework hey you create the object.
         3. Dependency injection
            1. Dependency

A class depending on B class object in context of a function

* + - * 1. Injection

Instead of we creating the objects the framework creates the objects.

* + - 1. Wiring
         1. Spring doing dependency injection.
  1. X+ spring = x becomes easier
     1. Spring will do the generic things
     2. We need to do the project specific things.
  2. Very vague ( AOP)
     1. Stick with expansion without knowing what it is Aspect Oriented programming.

1. Spring boot main feature
   1. It will autoconfigure, when compared to spring the amount of time we spend configuring is low in spring boot
2. Insulting an architecture by tying to a particular technology.
3. Practically when we want to work on spring.
   1. We should know architecture
      1. Entry point, exit point
         1. Excel sheet
      2. Hard things in a project
         1. Do we know what will change, what will not change.
4. Small things which can kill in the project.
   1. Being good at variables and functions..
   2. When we define a function
      1. Think who is going to call it
      2. How will the guy know things succeeded or failed , failed because of user , independent of user…
5. Generalize our learning..
   1. So that we don’t tie ourselves to particular technology. Less things to remember.. way to tell we are using “science”
   2. Example
      1. Interface
      2. Annotation
         1. Most of times we the users of annotation.
         2. Then it means we know the benefit we get and when.
         3. Start using the annotation and check did you get the benefit.
      3. All spring applications
         1. We are going to create classes and interfaces.
         2. We will tell spring hey create the objects and wire it.
         3. Then we will ask spring hey give me this object.
         4. Once you get object.. call non static function
6. While working on spring
   1. Core java ghost will attack big time and make the person tell spring is not working.