* Package based development
  + A practice of working with condensed nuggets of metadata contain in related groups
  + We group things that go together
    1. Example: All of the components of a feature
  + We can group things that depend on similar things
  + Packages are all related data and functionality that goes into that package
  + Why use it?
    1. Installation of the packages are really easy
    2. Can utilize version control systems
       - You are creating versions of the packages
  + Downsides
    1. Need to be very aware and thoughtful in what metadata goes where
    2. Requires a lot of structure and forethought
  + Unlocked packages
    1. In PBD, we do not use managed or unmanaged
       - It is unlocked
    2. Not meant to be distributed
    3. Strictly versioned groups of related metadata used to migrate customizations
       - These are meant to be distributed to orgs that we control
  + How we do this
    1. You need to set your org up as a Dev Hub
       - Just another word for your org
       - The org that your packages are created off of
       - It is the only place our packages can be versioned from
       - Steps
         * Navigate to setup
         * Search Dev Hub
         * Enable Dev Hub
         * Enable unlocked packages and second generation managed packages
    2. Create a new project in vs code
       - Not with manifest
    3. Authorize a Dev Hub
    4. Create a scratch org
       - This is a temporary blank org for development use
       - By default they last for 7 days
         * You can make it last from anywhere from 1-30 days
       - You can view them from the app launcher
       - Steps
         * Open vs code in your project
         * Create a new scratch org
         * Add an alias
         * Set the number of days
    5. Create a new package
       - Navigate to vs code terminal
       - sfdx force:package:create -n CustomPackage -t Unlocked -r myNewFolderContainingPackage -d “This is a description”
         * Note that myNewFolderContainingPackage must be a folder you create before running the command
    6. Add information to the package
       - Create your new data to add to the package
       - Set the path (through sfdx creation) to your package
       - You may need to add a main and default folder if not already there
         * Also add the folder name of component you are adding (like classes, aura, etc.)
    7. Change the version of the package
       - Sfdx force:package:version:create -p CustomPackage -x optionalPassword -w waitTimeInMinutes
    8. Pushing/pulling dev changes
       - Do as much development in vs code as is possible
       - Pushing takes your customizations to the scratch org from vs code
       - Pulling takes your customizations from your scratch orgs and adds them to vs code
       - Push to default scratch org
       - Delete force-app package so that we do not pull or push default
         * Also delete the force app folder
       - Be sure to set the default package to the one you have created
       - Force ignore file will not push or pull any file in that file
       - Pull source from default scratch org
  + Dependencies and how you can create them
    - These are how we denote that one package needs other packages to be installed before it can be installed
    - Add a section under default
    - “dependencies”:
      * “package”: “CustomPackage”,
      * “VersionNumber”: “”
    - Add a .NEXT or a .LATEST in your dependencies at the end of the version to act as wildcards
    - If a package depends on a package that depends on a package but the first one needs to use something from the last one, a dependency needs to be added for the last one