**Panel Study Guide**

**Java**

**What are the 4 pillars of OOP and examples of each?**

* **Polymorphism**
  + Allowing an object to take on many forms using overriding and overloading.
  + Can pass more than one Is-A test
  + When a parent reference is used to refer to a child class object
* **Encapsulation**
  + Used to hide important data known as **data hiding**
  + Making fields private and using getter methods to retrieve those fields
  + Wrapping fields and methods into one class
* **Abstraction** 
  + Pulling away complexity from a particular object for simplicity and reusability
  + Hides exactly “how” the object does it but the user knows “what” is does
* **Inheritance**
  + Where a child class(sub) inherits all methods from its parent class(super)
  + Done in a hierarchical order.

**Difference between overriding vs overloading**

* **Overloading** 
  + Same method name but different argument list within the same class
* **Overriding**
  + Same name, same return type and same arguments within two different classes which are the super class(parent) and sub class (child)

**JVM vs JRE vs JDK?**

* **JVM (Java Virtual Machine)**
  + Loads code, verifies code, executes code and provides a runtime environment
  + Can run on multiple types of platforms
* **JRE (Java Runtime Environment)**
  + Provide a runtime environment
  + It is the implementation of JVM and contains a set of libraries
  + JRE = JVM + library classes
* **JDK (Java Development Kit)**
  + Contains JRE and dev tools
  + JDK = JRE + dev tools

**What is the difference between Final, Finalize and Finally?**

* **Final**
  + Final is a keyword that can be used on classes, methods and field
    - Final fields cannot be reassigned
    - Final methods cannot be overridden but inherited
    - Final classes cannot be extended or inherited
* **Finally**
  + The finally keyword is used at the end of a try catch block and is optional. It is declared after the try/catch is called regardless if it is caught. It is always executed.
* **Finalize** 
  + It’s a method that is called for an objects destruction by the garbage collector
  + It is used to ensure an object terminates cleanly

**What is an array?**

* a grouping of objects under a common name
* can do any data type of an Array
* e.g. String[]

**What is Autoboxing, boxing and unboxing? When introduce in Java?**

* **Autoboxing** 
  + - The automatic conversion of a primitive datatype to its corresponding wrapper class
* **Boxing**
  + - Converting a primitive datatype to an object
* **Unboxing** 
  + - Converting an object into its corresponding datatype

**What does the static keyword do?**

* Static variables stay for the duration of the program whereas instance is created, used and destroyed after they’re done with use.
* The JVM reads the static keyword and loads them before anything else in the program
* Don’t need to create an object reference because we can just use the class name

**What is a local variable's default value?**

* They do not get a default value

**What is the Java Collection Framework? Name 5 methods found.**

* **Iterable**
* **Collection**
* **List**
* **Set**
* **Queue**

**What is the difference between Error and Exception?**

* **Error**
  + Considered fatal and unrecoverable
* **Exception** 
  + Not fatal and recoverable

**Checked vs Unchecked?**

* + Unchecked
    - Occurs at runtime and can be resolved with throws keyword or a try/catch block
  + Checked
    - Occurs at compilation and is mistakes done by the developer and bad coding practices.

**What is the root class or interface of exceptions?**

* You have the RuntimeException that extends Exception that then extends Throwable which extends to Object.
* Exceptions is the base class of all exceptions

**What is Serialization?**

* Serialization is when an object is converted to byte stream

**What is multithreading?**

* Executing many threads at once

**How can we spin up a new thread?**

* Create a thread sub class and override a run()
* Pass an object that implements runnable()

**What are some thread methods?**

**What are wrapper classes? Why are they needed?**

* Used to convert primitive data types into objects so the Object class may use it

**Threads join method?**

* Allows for multiple threads to execute only one at a time. Until the first thread is done executing then the next thread will not execute

**Why are strings immutable?**

* **Security** 
  + If string were mutable then the string name that the information was stored in could easily be changed
* **Class loading** 
  + Mutable strings would lead to the wrong class being loaded with another class argument
* **Synchronization** 
  + Mutable strings would lead to threads being unsafe

**Are you familiar with running Unix commands from Java?**

* **Scp** copies a file from one host to another
* **df**  checks for current space with your hard drive

**What's the difference between equals method and hashcode method? ==?**

* **equals()**
  + Checks to see if the object pass as the argument is equal to the object the method was invoked
* **hashcode()**
  + returns a hash code value as an integer and benefits hashing based classes such as hashset, hashmap and hashtable.
* **==**
  + Compares the references of the 2 objects

**What is encapsulation and its benefits?**

* Allows for protection of important information by making fields private and using public methods to access them which is known as data hiding.
* We can modify the code without breaking it which allows for reusability, maintainability and extensibility

**Differences of the this and super keyword?**

* **This**
  + You can refer to any member of the current object from within an instance method or constructor
* **Super**
  + Can use the super keyword can be used to access an overridden method in the parent class

**What are the variable scopes?**

* Block, method, object and class

**What are wrapper classes**

* Used to take a primitive variable and change it into an object

**Can a static method call a non-static method?**

* No, it cannot
* a non-static variable cannot be referenced from a static context

**Can interfaces have concrete methods? Can they have variables?**

* Yes after java 1.8. yes the variables are implicitly public static final

**What is the Java Collections framework?**

* It is a framework of interfaces and classes that group individual objects as a single unit

**Name 5 methods found in the Java Collections.**

* **Set**
  + Doesn’t allow duplicates
* **List** 
  + Allows for duplicates, and is ordered
* **Queue**
  + Ordered FIFO (first in first out)
* **Deque** 
  + Can be both FIFO and LILO (last in last out)
* **Map**
  + Contains key value pairs, no duplicates

**What are the differences of hashsets and treesets**

* **Hashsets**
  + Faster, unordered
* **Treesets** 
  + Slower, sorted order

**How do we create a checked exception?**

* Create a class that extends Exception

**How do we create an unchecked exception?**

* Create a class that extends RunTimeException

**What is reflection?**

* It is an API used to examine or modify classes, methods and interfaces at runtime

**What is a java bean?**

* It is a pojo that has private fields, public getters n’ setters, no args constructor and an overridden toString()

**What does transient mean?**

* It is a keyword that is used to mark a variable to not be serialized

**What are two ways of spinning up a new thread?**

* Creating a class that Implements the RunnableInterface, easiest way
* Creating a class by extending the Thread case and overriding the run()

**Difference between run() and start()**

* **Run()**
  + Entry point for the thread
* **Start()**
  + Starts a thread by calling its run()

**Difference between singleton and factory design patterns**

* Singleton can only be instantiated once and can be changed. Factory can instantiate a new object every time you need to access its variables

**Is there a limit to the size of an array?**

* Yes, but it is a very high number like 2 million

**Main method**

* Public static void main(String[] args) {}

**What are Instance Variables and example code?**

* Declared within a class and not inside a method
* class Taxes {int income;}

**What are Local Variables and some example code?**

* Declared within the method
* int Taxes() {int income}

**Method overloading code example**

Public int multiply(int a, int b) {

Return a \* b;

}

Public int multiply(int a, int b, int c) {

Return a\*b\*c;

}

**Overriding**

class Animal {

public void move() {

Sysyem.out.println(“animal moves”);

}

}

Class Dog extends Animal {

Public void move() {

System.out.println(“dog moves”);

}

}

**Can you have more than one main method?**

* Yes, they can be and could be used for primitive testing but Junit has replaced that.

**Junit**

**What is a unit?**

* A unit is small piece of your applications

**What is Junit?**

* it is framework within spring for test driven development of an application

**What are the Annotations of Junit?**

* @beforeclass
* @before
* @test
* @after
* @afterclass

**Difference between @Before and @BeforeClass?**

* **@Before**
  + Is initiated every time a @Test is used within the class
* **@BeforeClass**
  + Is initiated only once at the start of the first test of the class

**SQL & JDBC**

**What is SQL?**

* SQL stands for structured query language which is a programming language used to create a relational database and manipulating them as such

**DML?**

* It is a sub language of SQL, stands for data manipulation language which allows us to manipulate the data using statements such as: SELECT, INSERT, UPDATE and DELETE

**DDL?**

* it is a sub language of SQL, stands for data definition language which allows us to define our tables using statements such as: CREATE, ALTER AND DROP and TRUNCATE

**TCl?**

* It is a sub language of SQL, stands for transaction control language which allows us to make and manage transactions within the DB using statements such as: ROLLBACK, SAVEPOINT and COMMIT

**Constraints?**

* Constraints are parameters enforced on the column of a table or table itself. The following list of constraints are:
  + Primary key
  + Foreign key
  + Not null
  + Unique
  + Index
  + Default
  + Check

**Referential Integrity?**

* It ensures the relationship between tables and prevents orphan records from happening

**What's the advantage of a view?**

* Allows you to simplify complex queries and look at only certain tables and columns
* Provides an extra layer of security by limit data access to certain users

**Database Normalization and the NFs you know?**

* Normalization is a technique where you limit data dependencies within a given table, has many forms (about 10 according to wikipedia) but the 1st three are:
  + 1st normal form – must be atomic meaning we must break it down into its simplest form, can use a primary key
  + 2nd normal form – must be in 1st normal form, removing partial dependencies, getting rid of a column that is acting like a primary key (candidate key)
  + 3rd normal form – must be in 2nd normal form, must remove transitive dependencies, table(a, b, c, d) and a->b, b->c, so a->c

**How would you see common data between 2 tables?**

* By using the JOIN statement

**What is a self-join?**

* It is a query that joined to itself to compare values in a column to values in that same column of the same table

**What are SQL triggers?**

* it is a stored procedure that is called when a statement is called e.g. triggers used with sequences to make sure a new record is inserted every time it is called

**What is a cursor in SQL?**

* It is a temporary workspace used to store the data retrieved and manipulated by the database

**What is JDBC?**

* It stands for java database connectivity which is an API written in java to connect and communicate to DBs

**Important Interfaces of JDBC?**

* Statement
* Prepared statement
* Callable statement

**Difference Between Statement, PreparedStatement, CallableStatement?**

* Statement can execute SQL statements
* Prepared statement (extends statement) can execute precompiled statements making them faster and more secure than statements
* Callable statement (extends preparedstatement) can execute stored SQL procedures where preparedstatement gives methods for IN/OUT parameters

**How to do a transaction in JDBC?**

* Auto commit mode is defaulted to enabled so it will complete 1 transaction per statement. Use con.commit() after your multiple statements to perform the transaction

**Describe ACID properties**

* **Atomicity** 
  + Each transaction is treated as a single unit
* **Consistency**
  + Ensures the transaction can bring the DB from one valid state to another
* **Isolation**
  + Ensures that even with concurrent transactions the DB is in the same state as it were even if they were done sequentially
* **Durability** 
  + Once a transaction is committed, it stays that way even if there is a failure within the DB

**Servlet & Http**

**List the process of communication**

* Client sends an HTTP Request the server(tomcat), the deployment descriptor catches it (web.xml) finds the right url request and goes to that servlet class, goes the service, goes to the dao, goes to the database, then way back with the response.

**What is a servlet?**

* A class that implements Servlet interface and catches HTTP requests and process them.

**Lifecycle of Servlet?**

* Init()
  + is initialized once
* Service()
  + Is called every time the servlet is called
* Destroy()
  + Can happen randomly, but called once at the end of the lifecycle

**What are some of the HTTP Methods?**

* POST, GET, PUT, DELETE, PATCH, OPTION, TRACE, HEAD

**What are some of the HTTP Status Codes?**

* 100s
  + information
* 200s
  + ok
* 300s
  + redirect
* 400s
  + Client-side errors
* 500s
  + Server-side errors

**What is the Servlet API structure? Aka inheritance tree, types? Class/interfaces**

* MyServlet extends HttpServlet which extends the Servlet Interface

**Write a resp directly from a Servlet?**

* Protected void doGet(HttpRequest request, HttpResponse response) throws IOException, ServletException {}
  + Browser sends a request to the server and the server sends a response back to the browser

**How do handle a servlet responses?**

* Printwriter, redirect and forward

**Difference between Redirect & Forward?**

* Redirect
  + Request is redirected to a different resource
  + Executed client side
  + Slower because it must go through the browser
  + URL changes after the redirect
* Forward
  + Request is further processed server side
  + Executed server side
  + faster
  + URL stays the same within the browser

**How do we manage a session in java?**

* Using a server (tomcat)

**HTML, CSS & JavaScript**

**Difference between HTML, CSS, JavaScript, and JSP**

* HTML
  + Stands for hypertext markup language which used to build and structure web pages
* CSS
  + Stands for cascading style sheets which is used to show the presentation layout, colors and fonts
* JavaScript
  + It is a loosely typed programming language used to create dynamic and interactive web pages for a better user experience
* JSP
  + Stands for java server page

**What are the new things in HTML5?**

* Has new tags such as section, header, footer, nav, ect.
* Has new inline element tags such as mark, time and meter
* Has new element tags like video and audio

**Difference between == and ===**

* == compares the values
* === compares the values and datatypes

**What is ECMAScript?**

* Stands for European computer manufacturer’s association which ECMAScript is a standard of scripting languages

**Describe how you would debug an application**

* Open the web developer tool, select the debugger, set breakpoints and run the web app and run through the program line by line

**How can we manipulate the dom?**

* DOM stands for document object model and we can use javascript to manipulate that

**What is the arguments object in JavaScript?**

* It is a special construct available inside all function calls, it represents a list of arguments, it acts like an array-like object

**What is semi colon injection?**

* It happens when you miss semi colons, so the program will inject those into where it’s need

**Differences between Java & JavaScript?**

* **Java** 
  + Is strictly typed language
  + Programming language
  + Can run on a web browser or the JVM
* **JavaScript**
  + Is loosely typed language
  + Scripting language
  + Can only run on a web browser

**What is JSON?**

* Stands for javascript object notation, is a file format that uses human-readable key value pairs and array data types to send.

**What are the benefits of using AJAX?**

* An increase in speed and performance, reduces traffic from client to server

**What is Use Strict?**

* It is a mode in JS to throw more errors for bad coding practices, places your app in a “strict” operating context

**What are valid scopes of a variable in JS?**

* Global and local

**What is the difference between push and pop methods in an array**

* **Push**
  + Pushed in one object at the end of the array
* **Pop** 
  + Pop sends one object out of the array

**Hibernate**

**What is Hibernate?**

* It’s an ORM framework in which we use to abstract JDBC, it allows us to map our classes to the tables within the database

**What are the benefits of Hibernate?**

* Database independent meaning we can use any DB, lessens the amount of SQL written, allows for automatic dirty checking and allows for level 1 & level 2 caching

**Most common interfaces of Hibernate?**

* Configuration, SessionFactory and Session

**How do you map a table to a class?**

* By using the @Entity annotation

**States of Hibernate?**

* **Transient**
  + When the object is created and not with any data
* **Persisted**
  + A representation with an identifier in the DB, within the session
* **Detached** 
  + The object is persisted, and the session is closed

**Save method vs persist method?**

* **Save**
  + Adding a new entity to the table and returning a serializable id
  + Saves the change to the DB outside the transaction
  + Original hibernate method
* **Persist**
  + Same as the save method but it persists the instance
  + Doesn’t save the change to the DB outside the transaction

**What kind of hibernate exception have you had and how did you fix it?**

* **QuerySyntaxException**
  + Forgetting to use the @Table annotation
* **UnknownId.generator**
  + naming the seq generator wrong within the @GenerationValue
* **LazyInitilizationException**
  + I tried to call a lazy loaded children object when the session was closed, switched the fetching to eager and it worked.

**What is Automatic Dirty Checking?**

* After an object has been modified it will check for these and update them automatically

**update() vs merge()?**

* **Update()**
  + Will update if there is no object is within the session
  + Original hibernate method
* **Merge()**
  + Use if you are unsure if the object in a certain state in the session

**Is the sessionfactory thread safe?**

* Yes, it is thread safe and can run many threads concurrently, the sessionfactory is immutable

**Why use JDBC over Hibernate?**

* Allows us to have more control as a developer, as for performance JDBC performs far better with large queries and DBs

**Properties of a Tx?**

* I don’t know

**Difference between Criteria and Query interface?**

* I don’t know the differences

**Lazy vs eager fetching, what is the default? Differences?**

* **Lazy** 
  + Returns the values when called upon
  + is the default for all relationships
* **Eager**
  + Pre-fetches the data so its ready to go

**What is JPA?**

* Stands for java persistence api, it is a programming interface that allows us to store data into a DB

**Spring**

**What is IoC?**

* Stands for inversion of control and gives more control back to the developer through the use of DI

**What is DI?**

* Stands for dependency injection which means that we allow for out application to be loosely coupled and one class doesn’t rely completely on another to work unless used by the @Autowired annotation
* Removes the new keyword

**Why use Spring?**

* Allows for enterprise scaling, fast application building, allows you to loosely couple modules through DI and replace them without harming other parts of the application

**Differences between applicationContext & beanfactory?**

* **applicationContext**
  + part of the IoC container
  + uses eager loading
  + creates all the beans when initialized
* **beanfactory** 
  + part of the bean container
  + uses lazy loading
  + creates the bean only when requested

**Lifecycle of a spring bean?**

* Instantiate, populate properties, set name, set factory, set applicationContext, beforePostProcessor, AfterProperty properties, custom init, afterPostProcessor, use, destroy, custom destroy

**What is the flow of MVC?**

* Client sends an http request to the server (tomcat), then heads to the deployment descriptor in which the dispatcher servlet handles the request and sends it to the handler mapper to process the request, heads back to the DS, sends it to the controller, goes to the service, goes to the dao, goes to the DB then heads all the way back to the DS, moves to the view resolver, heads back to the DS and then goes to the view with the model back to the DS

**What is the Handler Mapper?**

* It is an interface that defines the mapping request and handler objects

**How can you handle exceptions in Spring MVC?**

* Use @ExceptionHandler or HTTP Status codes with @ResponseStatus

**How do you integrate Spring with ORM?**

* Spring data

**What is Spring Boot and why use it?**

* It is a framework that allows us to build stand alone applications really quickly.
* No need for war files when tomcat comes configured with it
* No XML config needed

**What is Spring Data?**

* It is a framework that allows for an easy programming model to access/ modifying data
* Object mapping tools and dynamic querying

**What are some Spring Data Repositories (interfaces)**

* CrudRepository, Repository, JpaRepository, PagingAndSortingRepository

**What are some annotations of Spring?**

* @Bean
* @ComponentScan
* @Autowired
* @Service
* @Repository
* @Configuration

**What are some annotations of Spring MVC?**

* @Controller
* @RequestMapping
* All CRUD mapping @s
* @PathVariable
* @RequestBody
* @ResponseBody

**What are some annotations of Spring ORM?**

* @Id
* @Table
* @Column
* @GeneratedValue
* @Entity

**Difference between beanwiring and autowiring?**

* bean wiring says here is an instance of a class and give me it when I need it
* auto wiring says please give me an instance of this class

**Difference between @Controller and @RestController?**

* @Controller
  + Required to use the @ResponseBody to make it a RESTful API
* @RestController
  + Automatically makes the responseBody Json

**What is AOP? Common uses of AOP?**

* Stands for aspect-oriented programming, used methods across all modules to create a new module, this new module can be: logging, transaction or security

**Difference between JoinPoint and PointCut in Spring AOP?**

* **JoinPoint**
  + different points in your application when a point is executing
* **PointCut**
  + Where exactly you can put your advice (code to be executed) and is a collection of multiple join points
* **Aspect** 
  + A module that encapsulates pointcuts

**Instantiate a bean container, 3 ways?**

* FileSystemxmlApplicationcontext
* XmlWebApplicationContext
* ClasspathApplicationContext

**REST**

**What is REST?**

* Representational State Transfer Architecture - An architecture, uses a WADL, all HTTP methods, XML and JSON, Documentation, HTTP Status Codes to handle exceptions - Rest in an architecture that makes a web service available for when users need it like a restful API

**What gets returned with REST?**

* Xml or json

**Have you heard of safe methods? Which ones?**

* GET, OPTION, HEAD

**Is REST a protocol?**

* No, it is architecture style used to define constraints to be used for creating web services

**What RESTful web services have you consumed and exposed?**

* My project APIs and the Chuck Norris API

**How would you protect a REST endpoint?**

* Using CORs filters

**What are imponent methods? Which are?**

* A method that can be called many times without changing the outcome of it for example GET & HEAD methods are imponent methods

**Why would you use REST over SOAP?**

* Uses HTTP crud methods, REST reads can be cached for better performance and scalability, supports many data formats that includes JSON and XML

**Marshalling vs Unmarshalling**

* **Marshalling** 
  + Turning a java object into a json object
* **Unmarshalling** 
  + Turning a json object into a java object

**What are some WSDL tags?**

* Definition – root tag
* Types -
* Messages - abstract message of the type
* Port types – what operations can be performed
* Binding – how the message is transfer
* Service – collection of related endpoints

**What some fault tags?**

* faultCode -
* faultString -
* faultActor – where the exception is

**What are the differences of HQL and Criteria?**

* **HQL** 
  + Returns the result set, can implement all CRUD methods
* **Criteria**
  + Returns a list, read only API

**Angular**

**What is Angular?**

* It is a javascript framework that helps build applications using HTML, CSS and typescript over JS.

**Benefits of Angular?**

* Gives applications a clean structure
* Includes lots of re-usable code
* Makes apps more testable

**What is typescript?**

* It is a strongly typed language that acts kind of like java that is transpiled back into javascript before its ran.

**What is NPM? Name some commands**

* Stands for node package manager which is used to hold all our node dependencies, some commands are npm install, npm start, npm –v

**Can you have multiple constructors in typescript?**

* Yes, but you can only have one implementation of it

**What are the lifecycle hooks of an Angular components?**

* ngOnInIt and directives have a lifecycle of creating it, rendering it, creating and rendering its children, updates and destroys it before removing it from the DOM

**How do you build your angular application?**

* Using the commands: ng new “project name”, ng generate “component name”

**What is an HTTP module?**

* Now we use HTTP client which handles HTTP requests

**What are decorators?**

* Attach meta data to classes or properties to modify data.

**What is the difference between and observable and a promise?**

* **Promise**
  + Returns a single value, not cancellable, not lazy
* **Observable** 
  + Work with multiple values, cancellable, we must subscribe to it, lazy

**How do you inject one html component into another?**

* Within the ts file of the component you can inject the root tag into whatever component you want it in

**What are pipes are used for?**

* Used to format data and how it is displayed

**How does angular protect against cross site scripting?**

* Angular treats all values as untrusted by default, can use DomSanitizer as trusted

**What is the difference between a service and a provider?**

* A service is a provider

**What is a provider?**

* A provider is an instruction to the DI system on how to obtain a value for a dependency. Most of the time, these dependencies are services that you create and provide.

**What is a Directive?**

* A class that modify the DOM, three types of directives: component directives, attribute directives and structural directives.
* **Structural** 
  + Modify the structure of the DOM, ngIf, ngFor, ngSwitch
* **Attribute**
  + Modify the attributes of DOM elements, ngClass, ngStyle

**What is an HttpClient?**

* what we use to make the http request and use http headers to pass the request

**Microservices**

**What is microservices?**

* Is a web service architecture that is used to create enterprise level architecture

**What are the different microservice patterns**

* Client sends http request -> api-gateway -> discovery service -> business service -> circuit breaking -> messaging que -> config server

**Benefits of microservices**

* Simple and small service can be taken down and build up easily, language agnostic, reduces single points of failure

**Drawbacks of microservices**

* Expensive, loses ACID gains BASE, will get complex

**What implementation have you used?**

* Zuul, hystrix, AWS SQS, spring cloud and configuration server

**What are the annotations?**

* @EnableEurekaServer, @EnableEurekaClient, @EnableZuulProxy, @CircuitBreaker, @HystrixCommand, @EnableConfigServer, @EnableConfigClient

**What are differences between application.properties and bootstrap.yml**

* Bootstrap loads first and application can override that

**What is Docker?**

* It is a containerization tool

**What is Docker Container?**

* The single snapshot of the application to use