Resources:

1. <https://stackoverflow.com/questions/34985805/difference-between-marker-interface-and-empty-interface>

**Iterator and Iterable**

**Design Patterns**

**Collections Interface**

* List
* Set
* Queue
* Collections vs Collection

Resources: <http://www.falkhausen.de/Java-8/java.util/Collection-Hierarchy.html>

**Comparator and Comparable**

* .equals() vs ==

Resources:

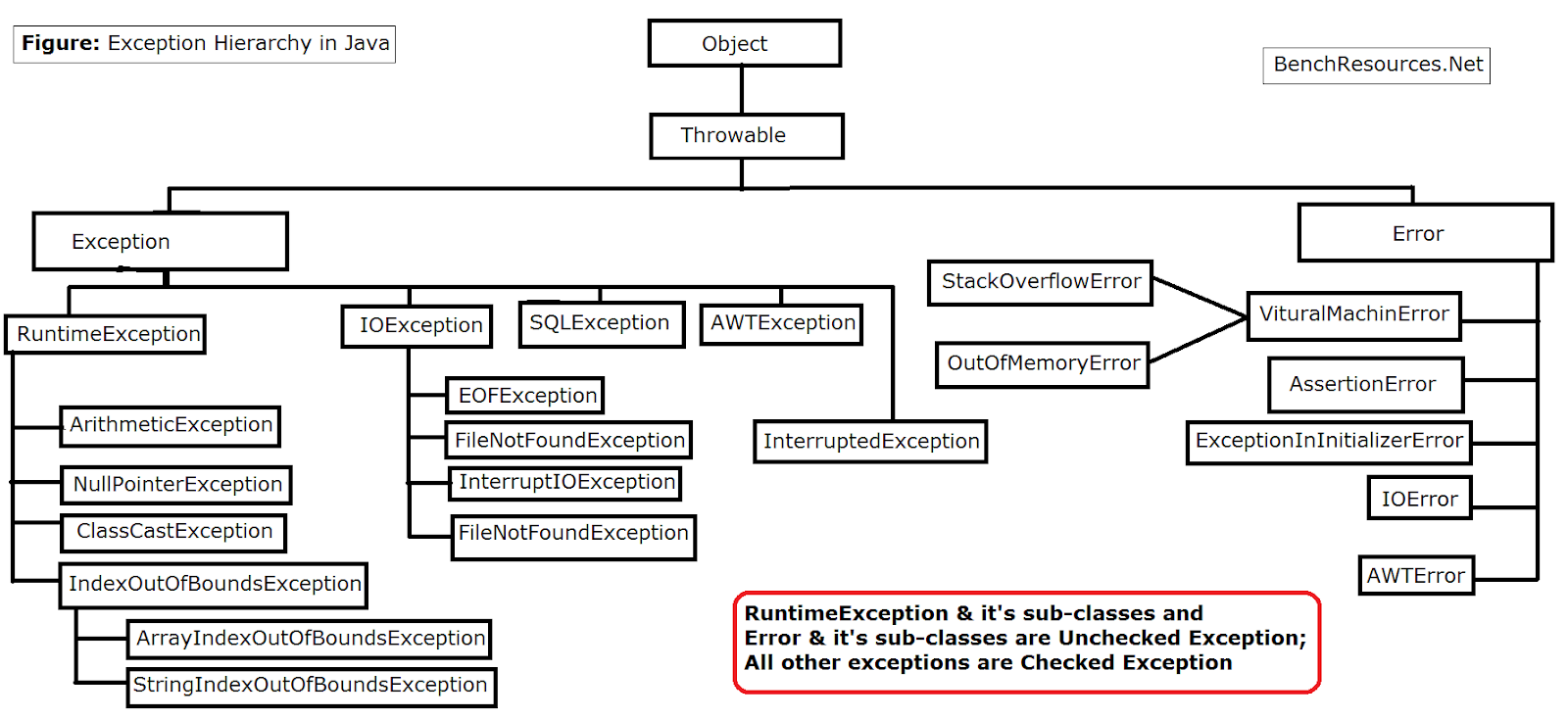
1. <https://www.geeksforgeeks.org/comparator-interface-java/>
2. <https://www.geeksforgeeks.org/comparable-vs-comparator-in-java/>

**Operators**

* Order of Operations

**Exceptions**

* Definition: In Java, Exceptions are objects that tell the application that something went wrong. They can be *thrown* because they implement the *throwable* interface.
* Handling
  + **try**/**catch**/**finally**
  + **throws**
* How are they different from Errors?
  + Though they implement the *throwable* interface, Errors are a different type of problem from Exceptions in that these are problems has to do with the system running the exception and fundamentally cannot be handled within the context Java.
* **throw** vs **throws**
  + **throw** is a keyword that is used to throw an actual exception from a method. You must throw your custom Exceptions for it to be handled somewhere.
  + **throws**,as stated before, is a keyword that handles exceptions by throwing it up the method stack.
* Checked vs Unchecked
  + Checked Exceptions are exceptions that only the programmer will understand how to handle and therefore these handling of these exceptions must also be implemented by the programmer.
  + Unchecked Exceptions are exceptions that are deemed as something the JVM can handle, therefore these do not have to be caught or handled by the programmer during implementation. These exceptions will be thrown all the way up the method stack and to the JVM itself.
* Hierarchy
  + Starts with the throwable interface, then errors and exceptions implements throwable, and then RuntimeExceptions and all the other checked exceptions extend Exception.



Resources: <http://www.benchresources.net/exception-hierarchy-in-java/>

**Threading**

* Thread Class
* Runnable Interface
  + A functional interface used by the thread class to execute desired functionality during a thread’s lifetime.
* Multithreading
* Consumer/Producer Problem
* Deadlock
* Dining Philosopher Problem

**Garbage Collection**

* A daemon thread that runs on the background of the JVM to destroys objects that are no longer referenced in memory and frees up space for further computational work.

**Binary, Octal and Hexadecimal**

* Base -
* Decimal -
* Binary -
* Octal -
* Hexadecimal -

**Streams**

**Key Terms**

|  |  |
| --- | --- |
| Term / Concept | Description |
| .equals() vs. == | Using the == operator with primitives will result in a boolean that indicates if the primitives being compared as the same, but using the same operator on objects will result in a boolean that indicates if the objects have the same reference. In those cases, it’s best to use the equals() method.  Scroll down [this link](https://stackoverflow.com/questions/767372/string-equals-versus) for a diagram of .equals() method vs == operator in action. |
| Abstract | The abstract keyword can be applied to classes and methods. Abstract classes may not be instantiated, but can contain abstract or concrete methods. Abstract methods do not have an implementation (meaning there should be no curly braces, even if there is nothing enclosed between the braces.)  The [oracle docs](https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html) go into more details about use-cases for abstract classes versus an interface. |
| binary/hex/octal  [Better Explanation Here!](https://medium.com/coderscorner/number-systems-decimal-binary-octal-and-hexadecimal-5e567e55ab28) | If decimal numbers are base 10 numbers then…   * binary is base 2 numbers (i.e: 0101010101…) * hexadecimals is base 16 numbers (i.e: 123456789ABCDEF0...) * octals is base 8 numbers (i.e: 12345670…) |
| Comparator & Comparable | [This link](https://www.javatpoint.com/difference-between-comparable-and-comparator) explains the differences very well, but the noteworthy differences are:   |  |  | | --- | --- | | Comparable | Comparator | | Must override compareTo() method | Must override compare() method | | Single Sorting Sequence | Multiple sorting Sequence | | Affects original class | Doesn’t affect original class | |
| Constructors | Special methods used to construct / instantiate the class it exists in. Keep in mind that the first line of constructors is always either super() or this(). |
| Consumer-Producer Problem |  |
| Deadlock | A situation where two or more processes are withholding resources from each other, making it possible for any processes to complete their task. |
| Default Methods | Default methods are used in interfaces to provide a concrete implementation of a method in an interface. This feature was not possible until the release of Java 8. |
| Default Constructors | Default constructors are constructors that don’t take in any arguments, and makes a call to the super class. Keep in mind that the first line of constructors is always either super() or this(). |
| Exceptions | Exceptions are events that occurs during execution, disrupting the normal flow of program instructions. When they occur, an exception object is generated which is thrown up the runtime stack to find a block of code to handle the exception. Please be acquainted with the exception hierarchy, which you can view [here](http://www.benchresources.net/exception-hierarchy-in-java/).  Keep in mind that Runtime Exceptions and Checked Exceptions are separated by a significant difference: handling is MANDATORY for Checked Exceptions, while it is OPTIONAL for Runtime Exceptions. This is primarily because Runtime Exceptions can happen anywhere anytime, while you can expect Checked Exceptions in certain situations (i.e: File I/O). Perhaps [this](https://www.geeksforgeeks.org/checked-vs-unchecked-exceptions-in-java/) article can help? |
| Error | An error is a subclass of Throwable; recovering from an error is not possible since they indicate a serious problem in the environment, so attempting to catch it makes little sense. |
| Final | This non-access modifier does different things depending where it’s applied…   |  |  |  | | --- | --- | --- | | Final Variables | Final Methods | Final Classes | | Cannot be reassigned | Cannot be overridden,  Can be overloaded | Cannot be extended | |
| Garbage Collection | Daemon thread (meaning it runs in the background) which cleans up objects with no references. |
| Git | Git is a distributed version control system used during software development to track and commit changes - pretty important when coordinating work with multiple programmers. |
| Instance | Instances (also known as objects) is a physical entity following the specifications of the class that it belongs to and contains state (instance variables) and behavior (methods). |
| Iterator/Iterable |  |
| Object | Objects (also known as instances) is a physical entity following the specifications of the class that it belongs to and contains state (instance variables) and behavior (methods). |
| Operators &  Order of operations |  |
| Primitive data types | Primitive data types include:  byte, short, int, long, float, double, boolean and char. |
| Runnable interface |  |
| Scope | ICBM!!! (Not intercontinental ballistic missiles) |
| Static | When applied to a variable, method, or class, that entity is shared among all objects of that class. Please note that static classes are always nested within another class, as no top-level static class exists in Java 8. |
| super() |  |
| this() |  |
| Thread class |  |
| Throw |  |
| Throwable |  |
| Throws |  |
| try/catch |  |
| try/catch/finally |  |
| try with resources |  |
|  |  |

**Week 2 - RDBMS, SQL, PL/SQL, JDBC**

**TO-DO:** {

Schema

Referential integrity

Orphan

Domain integrity

Constraints

Candidate keys

Primary key

* Natural keys, surrogate key

Foreign key

Composite key

Junction table - a table that allows us to represent a n:n relationship

Lookup table

Multiplicity - relationship between two entities,

Views

**PL/SQL - Procedural Language Extension of SQL**

* What is it? Oracle’s flavor of SQL.
* **Sequences** (aka “Autonumbers”) - Objects in oracle used to generate a number sequence
* **Triggers** - stored procedure that fires when an event occurs
* **Stored Procedures** 
  + Block of executable code that has full DML and TCL capabilities. Can take any amount (0 to Many) of IN and OUT parameters.

|  |
| --- |
| CREATE OR REPLACE PROCEDURE (name) [list of params] {IS/AS} [declare params] BEGIN [code to execute (use null; for empty execution)] EXCEPTION [exception handling] END; / |

* **Functions** 
  + Blocks of executable code which RETURN 1 value (which can be a cursor, representing a result set FYI). Can also have 0 or many both in and out parameters. Use DQL only! No insert, update, or delete capabilities.
  + Invoke via execute functionName();
* **Cursors** 
  + Oracle creates a “context area” with all information necessary for processing a SQL statement
  + A cursor is a pointer to that context area
  + There are implicit and explicit cursors
  + Implicit cursors are created by oracle whenever we execute a DML statement (and no explicit cursor exists). We have no control over implicit cursors. ***(examples needed)***
  + Explicit cursors are programmer defined, used to obtain more control over a context area ***(examples needed)***
    - *DECLARE* cursor to initialize memory
    - *OPEN* cursor to allocate memory
    - *FETCH* cursor to retrieve data
    - *CLOSE* cursor to deallocate memory

**TRANSACTIONS**

* Unit of work done on a database that may include many operations and is persisted with a commit.
* Properties of a Transaction **(ACID)**

1. Atomicity (aka “all or nothing”) - either the entire transaction executes successfully, or no commit is made
   1. Consistency - database remains in a valid state (according to existing structure and constraints) after each commit
   2. Isolation - the system state during concurrent transactions is the same as if the transactions were sequential
   3. Durability - all commits are final and cannot be rolled back, even in the case of system failure

* When dealing with isolation, there are three major problems that can occur in databases.
  + Dirty Read:
    1. Session 1 begins transaction and modifies data
    2. Session 2 begins TX and sees S1’s uncommitted data.
    3. Session 1 rollback uncommitted data! Basically never technically existed. Now S2’s data is no longer valid
  + Non-Repeatable Read:
    1. Session 1 begins TX, retrieves row of data
    2. Session 2 updates the row and commits the changes while S1 is still in progress
    3. Session 1 attempts to access the same row but cannot anymore
  + Phantom Read
    1. Session 1 begins TX executing a query
    2. S2 inserts data matching the query
    3. If S1 re-runs the query, “phantom data” has appeared.
* The **Isolation Levels** prevent different problems

|  |  |  |  |
| --- | --- | --- | --- |
| **Isolation Levels** | Dirty Read | Non-Repeatable Read | Phantom Read |
| Read Uncommitted | Allowed | Allowed | Allowed |
| Read Committed (default) | Prevents | Allowed | Allowed |
| Repeatable Read | Prevents | Prevents | Allowed |
| Serializable | Prevents | Prevents | Prevents |

**JDBC**

* Java Database Connectivity
* Connection
* ResultSet
* DAO design pattern
* *Compare the three Statement interfaces.*
  1. Statement
  2. PreparedStatement
  3. CallableStatement
* *What is SQL injection?*
* *How can i call a stored procedure from Java?*

**QC Questions**

1. What is the difference between order by and group by?

Order by sorts the order which the results are returned, while group by will aggregate records by specified columns.

1. Why would I denormalize a database?

To introduce redundancy...to make the database more readable.

1. Difference between implicit and explicit cursors?

Implicit cursors are automatically created for you by Oracle when you execute a query. Explicit cursors are cursors you create yourself.

1. What are the transaction isolation levels and what anomalies do?

They Read Uncommitted protects against nothing, Read Committed protects against dirty reads, Repeatable Reads protect against Nonrepeatable Reads and Serializable protects against Phantom Reads.

1. Explain multi-version concurrency control and what it is used for.

Multi-version concurrency control deals with the idea of multiple users keeping concurrent access to the database. This is kept with transaction isolation levels. Note to anyone: if someone asks you this, tell Genesis and she’ll probably fight them (to the death?).

1. What is the difference between clustered and unclustered indexes?

A clustered index spans across multiple columns while an unclustered index is for one column of data.

**Week 3 - HTML, JavaScript and HTTP**

**Interview Questions**

1. What is the difference between including JS in the <head> tag vs at the bottom of the <body> tag?

Generally, the javascript code will function in the head before the code in the body. The head section is usually used to contain information about the page that you don't necessarily see like the meta keywords meta description or title of a page. You would also link to any external files like .css .js files in the head section as they need to load before the page is displayed.  
  
Anything in the body section is what you would expect to be seen on screen.

**JavaScript in the head section** is usually ***meant to preload certain files*** (usually procedures or functions as the case may be). For example, a website that utilizes the Time() or Date() function would require that the .js file that contains those functions be called before the website is fully loaded allowing the instance to be available (preloaded) before imminent usage. Same also applies to other custom functions.

**JavaScript in the body section** is primarily for ***adding extra functionality*** to a website. An example is in the case of a custom .js file where a function is to check for correctness of words in an input string or matching all characters entered in an input string to be of a certain length.

1. What is JavaScript?

JavaScript is a lightweight, interpreted programming language with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages

1. What is AJAX?

AJAX stands for Asynchronous JavaScript and XML. It is a group of related technologies used to display data asynchronously. In other words, it sends and retrieves data without reloading the web page.

1. What are the steps to send an asynchronous request with JavaScript?

Create XHR variable for HTTP REQUEST / define onreadystatechange/ open request / send request.

1. What does it mean for a request to be asynchronous?  
   Asynchronous means that the script will send a request to the server, and continue it's execution without waiting for the reply. As soon as a reply is received a browser event is fired, which in turn allows the script to execute associated actions.  
     
   Ajax knows when to pull data from a server because you tell it when to do it.
2. Explain type coercion.

Type coercion is the process of converting a value from one type to another (such as string to a number, object to boolean, and so on). Any type, be it primitive or an object, is a valid subject for type coercion

1. What is truthy and falsy?

In JavaScript, a truthy value is a value that is considered true when encountered in a Boolean context. All values are truthy unless they are defined as falsy (i.e., except for false, 0, "", null, undefined, and NaN).

1. What are the guard and default operators?

We can use the logical ‘AND’ (&&) and the logical ‘OR’ (||) when evaluating values. This can be useful for writing simple and concise return statements and variable assignments. The logical AND operator works like a guard

1. How can I inject variables into strings in JavaScript?

Using backticks, tilde, or ``.

1. What is closure? (<- what is this?)

An encapsulation technique in JavaScript that utilizes a nested function to keep access from the parent scope. It has access to all of the outer function's variables and input parameters. The closure also retains access to these variables even after the parent scope has returned.

1. What is a callback function?

A function that is passed to another function as a parameter.

1. Explain hoisting and scopes in JS.

Hosting means an undeclared variable is automatically considered to be in the global scope.

There is Local Scope where the variable is declared with the keyword let. Block scope refers to a variable within a block. That variable is declared with the keyword var. Global scope means the variable can be accessed anywhere.

1. What is jQuery and what is it used for?

* jQuery is a lightweight, "write less, do more", JavaScript library.
* The purpose of jQuery is to make it much easier to use JavaScript on your website.
* jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish and wraps them into methods that you can call with a single line of code.
* jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

1. What is JSON?

JSON is short for **JavaScript Object Notation** and is a way to store information in an organized, easy-to-access manner. In a nutshell, it gives us a human-readable collection of data that we can access in a really logical manner.

1. How is a JSON string converted to a JS object and vice versa?

JSON.parse to convert the string into a js object. Stringify JS object to JSON string

1. What is HTML? What is the root tag/parent element?

* HTML stands for Hyper Text Markup Language
* HTML is the standard markup language for Web pages
* HTML elements are the building blocks of HTML pages
* HTML elements are represented by <> tags
* The root parent of all html elements is the <html> tag

1. What is CSS? Where can I include it in my HTML?

* CSS stands for Cascading Style Sheets
* CSS defines how the html page is presented in the browser
* CSS can be included in the <head> and/or <body> tags

1. What is Bootstrap? Explain its grid system.

A framework that helps you style your web page easily

1. What is HTTP?

HTTP means HyperText Transfer Protocol. HTTP is the underlying protocol used by the World Wide Web and this protocol defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands.

1. What is the difference between a GET and POST method?

Both GET and POST method is used to transfer data from client to server in HTTP protocol but the Main difference between POST and GET method is that GET carries request parameter appended in URL string while POST carries request parameter in message body which makes it a more secure way of transferring data from client to server in HTTP protocol.

1. Explain event propagation? How do I stop an event from propagating?
2. What is DOM?

Document Object Model - DOM is a standard interface by which we can access/ update /add HTML elements.

1. What is an API?

In simple words, API stands for **Application Programming Interface.** API acts as an interface between two software applications and allows the two software applications to communicate with each other. API is a collection of software functions which can be executed by another software program.

1. What is an SDN? (Software Defined Network? Or CDN - Content Delivery Network?)

AWS is an example of SDN

1. How do I change the text in the tab of an HTML page?

<head><title>Text To Change Tab</title></head> Tags

1. What is an iframe?

Iframe stands for Inline Frame. The ” iframe ” tag defines a rectangular region within the document in which the browser can display a separate document, including scrollbars and borders. An inline frame is used to embed another document within the current HTML document.

1. What type of inheritance does JavaScript support?

Javascript supports prototype inheritance and every JS object has a property called \_\_proto\_\_ which points to the object’s prototype. This means that every JS object inherits directly from other objects.

1. What is ECMAScript?

* ECMA stands for European Computer Manufacturer’s Association.
* ECMAScript is a Standard for a scripting language.
* Languages like Javascript are based on the ECMAScript standard.
* ECMA Standard is based on several originating technologies, the most well known being JavaScript (Netscape) and JScript (Microsoft).

1. What are some of the properties of the XMLHTTPRequest object?
2. What are some of the methods n of the XMLHttpRequest object?

Week 4

* What is a servlet?
  + special CLASS that services HTTP requests for the server (container)
  + extends HttpServlet
  + configured in web.xml of the web app project
  + life cycle managed by "Web Container" (also called servlet container)
* What is the hierarchy of servlets?
  + Servlet (interface) all methods needed to make a servlet
  + GenericServlet (abstract class) implements most of the basic servlet
  + HttpServlet (abstract class)
  + MyServlet override methods you need
* What is the lifecycle of a servlet?
  + init()
    - Called 1 time when the servlet is first requested unless otherwise specified with <load-on-startup> tag - initializes when a page is rendered
  + service()
    - Called multiple times, whenever it is requested
    - usually NOT overridden
  + destroy()
    - Called once, typically when the server is being shutdown occurs
    - May override to clean up resources
* Describe the deployment descriptor
  + Web.xml
  + File in which we configure our web apps
  + Indicates which URL-patterns(endpoints) will map requests to specific servlets and configure other details like parameters and when to initialize our servlets
  + tells the web container where your servlets are, and how to map them
  + Mapping
    - taking one servlet and mapping it with a URL
* What are some key tags in the deployment descriptor?
* How do I map a Servlet?
* What are the doGet and doPost methods?
* What is the method signature of doGet and doPost?
* Describe the flow of a request with servlets
* What are the different ways I can send back a response?
* What is the difference between a redirect and a forward?
* What is the PrintWriter?
* What is ServletContext and ServletConfig?
* What is the difference between a GenericServlet and an HttpServlet?
* What is Angular?
  + It is a TypeScript-based open-source front-end web application framework.
* What is the Angular CLI?
  + Angular cli is a command line interface to scaffold and build angular apps using nodejs style (commonJs) modules.
* Where is the app folder located? What files does it contain if the project was created by the Angular CLI?
  + The app folder is located within the project's src folder. It contains files like app.module.ts, app.component.css, app.component.html, app.component.ts
* What is a decorator?
  + It is a special kind of declaration that can be attached to a class declaration, method, accessor, property, or parameter. It allows us to decorate classes and functions, similar to annotations in java and decorators in python.
* What is a module?
  + In Angular, a module is a mechanism to group components, directives, pipes and services that are related, in such a way that can be combined with other modules to create an application. An Angular application can be thought of like a puzzle where each piece (or each module) is needed to be able to see the full picture.
* What is a component?
  + Components are the most basic UI building block of an Angular app.
  + One piece of functionality you want to show up on your page
* What is @component? What are some attributes you can find inside the decorator?
  + It is a decorator that is added to the top of the component's class definition inside its component.ts file. Some attributes are the selector, templateUrl, and stylesUrls.
* How would you add a new component without the Angular CLI?
  + Create the component.ts file inside the src folder.
  + Add the "@component" decorator
  + Import the component into the app.module.ts file.
  + Add a reference to the imported component into the "@NgModule" declarations.
* What is a directive? What are the different types?
  + They are custom markers on a DOM element that tells Angular to attach behavior to the element.
  + Special pre-built instructions that typically render HTML code or provide additional functionality
  + The types are structural directives, attribute directives, component directives, custom directives using @Directive decorator
  + Structural: add/remove or update the contents of the DOM - \*ngIf, \*ngFor
  + Attributes: can change the look and feel or behavior of an element - ngModel, ngClass, ngStyle
* What is data binding? What are some different types?
  + Data binding is a core concept in Angular and allows to define communication between a component and the DOM, making it very easy to define interactive applications without worrying about pushing and pulling data. There are four forms of data binding: interpolation {{value}}, property binding [property]="value", event binding (event)="function", two-way data binding [(ngModel)]="value".
* What is a pipe in Angular? How do you create a custom pipe?
  + It is a way to write display-value transformations that you can declare in your HTML. You create a custom pipe by using the @pipe() on a class, implementing the PipeTransform. To you use it, you must also add it to a module.
* What is a SPA?
  + Single-Page Applications (SPAs) are Web apps that load a single HTML page and dynamically update that page as the user interacts with the app. SPAs use AJAX and HTML5 to create a fluid and responsive Web apps, without constant page reloads.
* What is a service in Angular?
  + Angular services are singleton objects which get instantiated only once during the lifetime of an application. They contain methods that maintain data throughout the life of an application, i.e. data does not get refreshed and is available all the time. The main objective of a service is to organize and share business logic, models, or data and functions with different components of an Angular application.
  + @Injectable
* What is routing? What is it used for?
  + Angular implements its own routing module in order to allow for single-page web applications. It is used to build SPAs.
* How do you perform HTTP requests in Angular?
  + Import the HttpClient module and include it in the ngModule imports.
  + Inject an object of type HttpClient inside of the constructor of the class you want to make an HttpRequest in.
  + Use the HttpClient property in your object and call one of the HttpRequest functions, such as get() or post().
* What is the difference between a promise and an observable?
  + Observables do not execute until subscription, while promises execute upon creation. Observables are cancellable, while Promises are not. Promises return only one value and observables return many.
  + Observable - a data structure that holds the response to an asynchronous request
* What is Dependency injection
  + services and injectable.
  + design pattern
  + common
  + allow a frame work, mange life cycle
  + don’t instantiate
  + use a decorator to call them.
* What is npm and node.js
  + Provides angular with a command line interface
  + Node
    - Server node javascript
  + Write server side js
  + Manage our dependies
  + Similar to maven- build tool
  + Manger of javasript depndies

**Angular**

1. What is Angular?

It is a TypeScript-based open-source front-end web application framework.

1. What is the Angular CLI?

Angular cli is a command line interface to scaffold and build angular apps using node.js style (commonJs) modules.

1. Where is the app folder located? What files does it contain if the project was created by the Angular CLI?

The app folder is located within the project's src folder. It contains files like:

app.module.ts, app.component.css, app.component.html, app.component.ts

1. What is a decorator?

It is a special kind of declaration that can be attached to a class declaration, method, accessor, property, or parameter. It allows us to decorate classes and functions, similar to annotations in java and decorators in python.

1. What is a module?

In Angular, a module is a mechanism to group components, directives, pipes and services that are related, in such a way that can be combined with other modules to create an application. An Angular application can be thought of as a puzzle where each piece (or each module) is needed to be able to see the full picture.

1. What is a component?

Components are the most basic UI building block of an Angular app.

1. What is @component? What are some attributes you can find inside the decorator?

It is a decorator that is added to the top of the component's class definition inside its component.ts file. Some attributes are the selector, templateUrl and stylesUrls.

1. How would you add a new component without the Angular CLI?

Create the component.ts file inside the src folder.

Add the "@component" decorator

Import the component into the app.module.ts file.

Add a reference to the imported component into the "@NgModule" declarations.

1. What is a directive? What are the different types?

They are custom markers on a DOM element that tell Angular to attach behavior to the element.

The types are structural directives, attribute directives, and component directives.

1. What is data binding? What are some different types?

Data binding is a core concept in Angular and allows to define communication between a component and the DOM, making it very easy to define interactive applications without worrying about pushing and pulling data. There are four forms of data binding: interpolation {{ value }}, property binding [property]="value", event binding (event)="function", two-way data binding [(ngModel)]="value".

1. What is a pipe in Angular? How do you create a custom pipe?

It is a way to write display-value transformations that you can declare in your HTML. You create a custom pipe by using the @pipe() on a class, implementing the PipeTransform. To you use it, you must also add it to a module.

1. What is a SPA?

Single-Page Applications (SPAs) are Web apps that load a single HTML page and dynamically update that page as the user interacts with the app. SPAs use AJAX and HTML5 to create a fluid and responsive Web apps, without constant page reloads.

1. What is a service in Angular?

Angular services are singleton objects which get instantiated only once during the lifetime of an application. They contain methods that maintain data throughout the life of an application, i.e. data does not get refreshed and is available all the time. The main objective of a service is to organize and share business logic, models, or data and functions with different components of an Angular application.

1. What is routing? What is it used for?

Angular implements its own routing module in order to allow for single-page web applications. It is used to build SPAs.

1. How do you perform HTTP requests in Angular?

Import the HttpClient module and include it in the ngModule imports.

Inject an object of type HttpClient inside of the constructor of the class you want to make an HttpRequest in.

Use the HttpClient property in your object and call one of the HttpRequest functions, such as get() or post().

1. What is the difference between a promise and an observable?

Observables do not execute until subscription, while promises execute upon creation. Observables are cancellable, while Promises are not. Promises return only one value and observables return many.

**Servlets**

1. What is a servlet?

Servlets are server-side software used to handle different requests and apply business logic wherever needed.

1. What is the hierarchy of servlets?

Servlet -> Generic Servlet -> HttpServlet (and other Servlets for different requests) -> (Your Servlet)

1. What is the lifecycle of a servlet?

First, it can initialized when the servlet is called the first time, when the init() method is called or it can be pre-initialized by including it in the <load-on-startup> tag in the web.xml. Then it services a bunch of requests and then it is destroyed when the destroy() method is called or can be either when the servlet context is ended. The destroy method ends the thread and sets the servlet up for garbage collection.

1. Describe the deployment descriptor.
2. What are some key tags in the deployment descriptor?
3. How do I map a Servlet?
4. What are the doGet and doPost methods?
5. What is the method signature of doGet and doPost?
6. Describe the flow of a request with servlets
7. What are the different ways I can send back a response?
8. What is the difference between a redirect and a forward?
9. What is the PrintWriter?
10. What is ServletContext and ServletConfig?
11. What is the difference between a GenericServlet and an HttpServlet?

What is the difference between node and npm?

1. https://www.quora.com/What-are-the-difference-between-node-and-nodejs-npm-vs-nvm

Item Potent – Repeatable results – send a request doesn’t effect server – post effects server- neither item potent nor safe – delete is item potent as I can delete in the server and if I keep sending a delete to the same part it won’t delete since the item is gone already.

Interview questions

* Collection and Collections
  + Collections is a utility class for the collection class

SQL

* Sublanguages of SQL
  + Data Definition Language - controls the structure or schema of the DB
    - CREATE, ALTER, DROP, TRUNCATE
  + Data Manipulation Language - interactions with the data. “CRUD”
    - INSERT, UPDATE, DELETE, SELECT
  + Data Control Language - specifying user permissions

GRANT, REVOKE

* + Transaction Control Language - organizing transactions - units of work with our data
    - COMMIT, ROLLBACK, SAVEPOINT
  + Data Query Language - querying data
    - SELECT
* JOINS
  + Cross join -
  + Natural join -
* DB Normalization
  + 1NF
    - Atomic, single value columns
    - Unique name columns
  + 2NF
    - No partial dependencies
    - No columns depending on primary key
  + 3NF
    - No transitive dependencies
* Multiplicity
  + One to many
  + One to one
  + Many to many
* PL/SQL - procedural language
  + Extension to oracle’s sql
  + Procedures do full tcl and dml
* Cursor
  + Mechanism by which you can assign a name to a select statement and manipulate information within it
  + Pointer to a context area, which is what oracle stores the results from a query in
  + Explicit are user defined
  + Implicit are oracle generated
* Index

Webpack – a bundleing build tool used for javascript frames. Module builder.

Maven used to build java application