Klassische Fragen für die Klausur

- **Was ist ein Konstruktor?**
  - Konstruktoren sind notwendig, um Objekte einer Klasse erzeugen zu können.
  - Wenn wir in einer Klasse keinen Konstruktor explizit implementieren, stellt Java uns automatisch einen parameterlosen Default Konstruktur zur Verfügung.
  - Eine Klasse kann mehrere Konstruktoren mit jeweils unterschiedlicher Parameterliste haben. Man spricht dann vom Überladen der Konstruktoren.
- **Was ist ein default-Konstruktor, und was passiert mit ihm wenn ich einen eigenen Konstruktor erstelle?**
  - Der Default-Konstruktor in Java
  - Wird für eine Klasse kein Konstruktor explizit angegeben, dann generiert der Java-Compiler automatisch den parameterlosen Default-Konstruktor.
  - Der Default-Konstruktor führt nur eine einzige Aufgabe aus. Er ruft den parameterlosen Konstruktor seiner Superklasse (Vaterklasse) auf.
  - Der Default-Konstruktor wird nicht erzeugt, wenn in der Klassendeklaration nur parametrisierte Konstruktoren definiert sind. In diesem Sonderfall besitzt die Klasse überhaupt keinen parameterlosen Konstruktor.
- **Was ist die main-Methode?**
  - Die Main-Methode in Java ist der Einstiegspunkt für die Ausführung eines Java-Programms. Es ist eine spezielle Methode, die von der Java Virtual Machine (JVM) aufgerufen wird, um das Programm auszuführen.
  - Es gibt einige wichtige Aspekte der Main-Methode, die beachtet werden müssen:
    - Die Main-Methode muss immer in der Klasse definiert werden, die als Einstiegspunkt für das Programm verwendet wird.
    - Die Main-Methode muss als public und static deklariert sein, damit sie von der JVM aufgerufen werden kann.
    - Die Methode akzeptiert ein Argument vom Typ String-Array, das args genannt wird und vom Benutzer bereitgestellte Argumente enthält.
    - Der Rückgabetyp der Main-Methode ist void, was bedeutet, dass sie nichts zurückgibt.
  - In der Main-Methode wird der Hauptteil des Programms ausgeführt. Sie kann andere Methoden aufrufen, Variablen deklarieren, Eingabe- und Ausgabebefehle ausführen und vieles mehr.
- **Wofür werden TUI´s genutzt?**
  - Wir genutzt um in Video Terminals den Cursor zu nutzen und Farben sowie Schriftarten zu unterstützen
  - Einfachere Bedienung der Kommandozeile (CMD, Powershell...)
- **Welche Möglichkeiten gibt es die Sichtbarkeit einer Klasse oder eines Paketes anzupassen?**
  - Public
    - Global sichtbar
  - private
    - Sichtbar nur innerhalb einer Klasse
  - protected
    - Sichtbarkeit für andere Klassen im selben package
    - Sichtbarkeit für abgeleitete Klassen

- package private
  - "package private" bedeutet, dass eine Klasse, ein Attribut oder eine Methode nur innerhalb des gleichen Pakets sichtbar ist und nicht von abgeleiteten Klassen außerhalb des Pakets

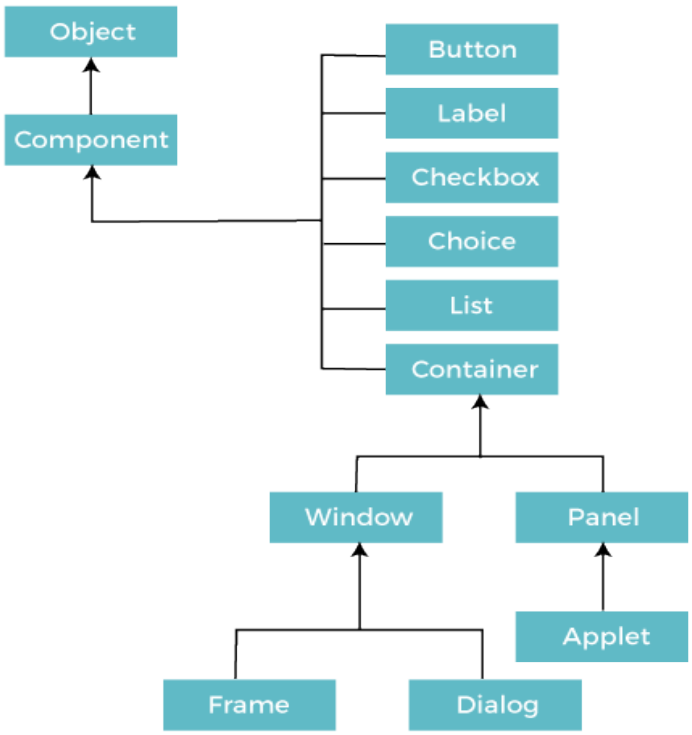
- **Was sind primitive/ komplexe Datentypen?**
  - Primitiv: Integer, Float, boolean, Double
    - Primitive Datentypen sind Datentypen, die so elementar sind, dass die Art und Weise sie dazustellen in Java eingebaut ist.
  - Komplex: Listen, Arrays, HashMap
  - Jede Klasse, die über mehrere Attribute verfügt, definiert einen komplexen Datentypen. Als komplex werden sie deshalb bezeichnet, weil sie sich im Gegensatz zu den primitiven Datentypen aus verschiedenen Elementen zusammensetzen.
- **Was ist Vererbung?**
  - Schlüsselwort: extends
  - Die Vererbung in Java ermöglicht es dir eine hierarchische Ordnung für Klassen festzulegen und reduziert somit die Menge des redundanten Codes. Das bedeutet, dass du die Attribute und Methoden einer Klasse vererben kannst und du sie damit nicht noch einmal programmieren musst.
  - Alle Klassen in Java erben direkt oder indirekt von der Java Basisklasse Object. Wird bei einer Klassendeklaration keine extends Klausel angegeben so wird die Klasse automatisch von der Klasse Object abgeleitet.
- **Was mache ich mit @Override und was ist das?**
  - @Override ist eine Annotation
  - gibt an, dass eine Methode in einer abgeleiteten Klasse eine Methode in der Basisklasse überschreibt. Es wird verwendet, um sicherzustellen, dass die signatur der überschreibenden Methode identisch zur signatur der überschriebenen Methode ist. Wenn die signatur der Methode in der abgeleiteten Klasse nicht mit der signatur der Methode in der Basisklasse übereinstimmt, wird ein Kompilierungsfehler generiert.
- **Was sind abstrakte Klassen?**
  - Eine abstrakte Klasse in Java ist eine Klasse, die nicht direkt instanziiert werden kann und mindestens eine abstrakte Methode enthält. Sie dient als Vorlage für abgeleitete Klassen, indem sie gemeinsame Methoden und Eigenschaften definiert, die von den abgeleiteten Klassen implementiert werden müssen.
- **Was ist eine abstrakte Methode?**
  - Eine abstrakte Methode hat keine Implementierung (bietet also keine Funktion)
  - Nicht definiert nur deklariert
- **Was ist ein Interface?**
  - Ein Interface ist eine Gruppe verwandter Methoden ohne Implementierung
  - Interfaces unterstützen Vererbung
  - Ein Interface in Java ist eine Sammlung von Methodensignaturen, die von einer Klasse implementiert werden können. Es definiert eine Verhaltensweise, die von einer Klasse implementiert werden muss, um als Teil einer bestimmten Schnittstelle zu gelten.
- **Was ist das Ziel der Oberflächenprogrammierung?**
  - Die Oberfläche soll:
    - intuitiv und einfach nutzbar sein
    - angenehme Nutzung für Nutzer
    - Effiziente Nutzung für User mit der Anwendung
- **Welche Varianten von Generics gibt es?**
  - Object-Variante

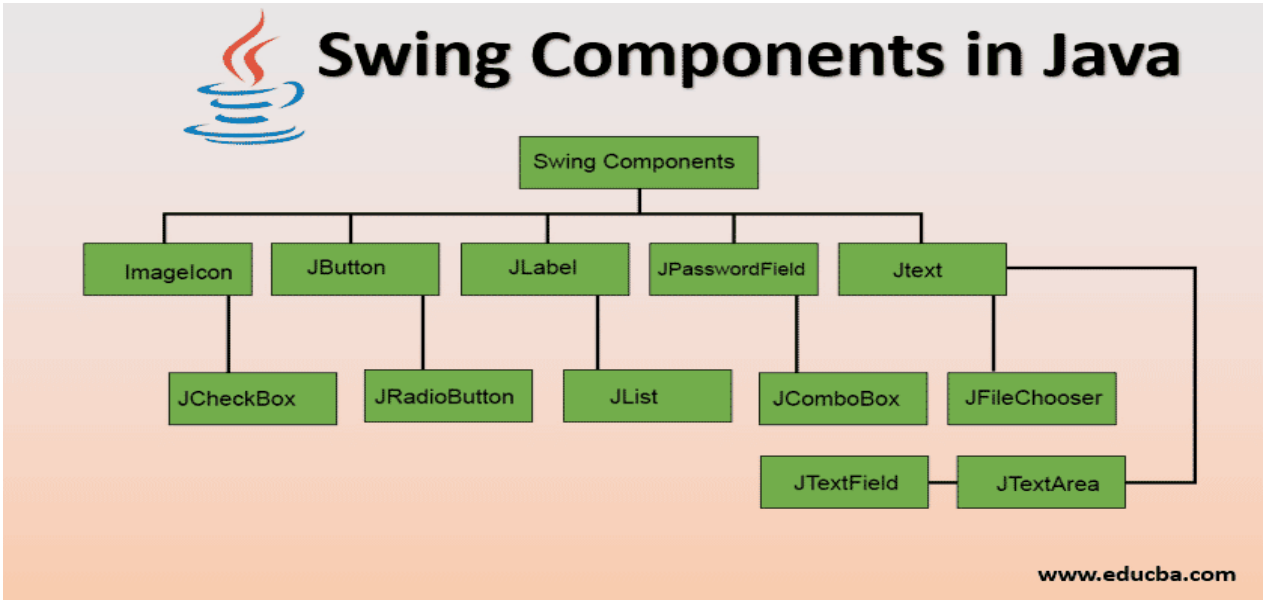
        - Klassen-Generics ermöglichen es, eine Klasse zu definieren, die mit verschiedenen Typen von Objekten arbeiten kann, ohne dass für jeden Typ eine separate Klasse definiert werden muss.
    - Typ-Parameter-Methode
        - Die Typ-Parameter werden in spitzen Klammern (<>) nach dem Klassennamen angegeben.
        - Methoden-Generics erlauben es, dass Typ-Parameter nur für eine bestimmte Methode definiert werden. Die Typ-Parameter werden in spitzen Klammern (<>) vor dem Rückgabetyp der Methode angegeben.
- **Was sind Standardlayouts vom Frame?**
    - Das Standard-Layout von Frame in Java ist das sogenannte Border-Layout. Dieses Layout teilt den Frame in fünf Bereiche auf: Norden, Süden, Osten, Westen und Mitte.
- **Was ist der Standard Layoutmanager bei paint (also Frames usw)?**
    - BorderLayout (default)
    - Für Dialog und JPanel ist es FlowLayout
- **Welche 2 Arten von Fenstern gibt es?**
    - Frames
    - Dialoge
    - Was ist der Unterschied?
        - Der Hauptunterschied zwischen ihnen besteht darin, dass Frame als eigenständiges Fenster fungiert, während Dialog als ein Dialogfenster fungiert, das in Bezug auf ein übergeordnetes Fenster modal oder nicht modal sein kann.
        - Ein Dialog ist ein Fenster, das in Bezug auf ein übergeordnetes Fenster modal oder nicht modal sein kann. Ein modaler Dialog blockiert die Eingabe von anderen Fenstern, bis der Dialog geschlossen wird, während ein nicht-modalen Dialog im Hintergrund weiterhin interagiert werden kann. Ein Dialog wird in der Regel verwendet, um eine Benutzerinteraktion zu ermöglichen, wie z.B. das Auswählen von Optionen oder das Eingeben von Daten.
- **BorderLayout was ist besonders?**
    - Center, → ist default (alles undefinierte landet da)
    - Center, nimmt den restlichen nicht benutzen Platz ein
- **Wichtig: Kevin steht auf super Konstruktoren. Was ist das?**
    - In Java wird das Schlüsselwort super verwendet, um auf die Methoden und Eigenschaften der Basisklasse zuzugreifen. Eine Super-Konstruktor-Methode wird verwendet, um einen Basisklassen-Konstruktor aufzurufen, wenn eine abgeleitete Klasse erstellt wird.
- **Was macht welches Layout – wie ordnet es die Komponenten an?**
    - BorderLayout
        - Teilt die Oberfläche in North, West, Center, East und South ein
        - Alle Komponenten die hinzugefügt werden und nicht explizit in einen Bereich zugeordnet werden landen im Center
    - CardLayout
        - Das CardLayout in Java ordnet seine Komponenten in einem Stapel an, wobei immer nur eine Komponente sichtbar ist.
    - FlowLayout
        - Das FlowLayout in Java ordnet seine Komponenten in einer horizontalen Reihe von links nach rechts an. Wenn der verfügbare Platz nicht ausreicht, um alle Komponenten in einer Zeile anzuzeigen, werden sie automatisch in die nächste Zeile verschoben. Das FlowLayout stellt sicher, dass die Komponenten horizontal

ausgerichtet sind und sich nicht überlappen.
    - GridLayout
        - Das GridLayout in Java ordnet seine Komponenten in einem Raster an, wobei jede Komponente in eine Zelle des Rasters platziert wird.
    - GridBagLayout
        - Das GridBagLayout in Java ist ein flexibles Layout-Manager, der es ermöglicht, Komponenten in einem Raster anzuordnen, wobei jede Zelle unterschiedliche Größen haben kann. Es bietet mehr Kontrolle über die Positionierung von Komponenten als das GridLayout, aber es ist auch komplexer zu verwenden.
- **Welche Layoutmanager muss man konfigurieren?**
    - GridLayout und GridBagLayout
    - beim GridBagLayout benutzen wir ein Hilfsobjekt → das GridBagContstraints Objekt
    - Regionen wo das Objekt angelegt werden soll werden über add() mitgegeben
- **Wege der Serialisierung?**
    - In Java gibt es zwei Wege der Serialisierung:
        - Serializable Interface: Das Serializable Interface ist ein Marker-Interface, das von Klassen implementiert werden kann, um anzuzeigen, dass ihre Objekte serialisierbar sind. Wenn eine Klasse Serializable implementiert, können ihre Objekte mithilfe von ObjectInputStream und ObjectOutputStream serialisiert und deserialisiert werden.
        - Externalizable Interface: Das Externalizable Interface ist ein weiteres Interface, das von Klassen implementiert werden kann, um ihre Objekte serialisierbar zu machen. Im Gegensatz zum Serializable Interface gibt es jedoch mehr Kontrolle darüber, wie die Objekte serialisiert und deserialisiert werden. Wenn eine Klasse Externalizable implementiert, müssen Sie die Methoden writeExternal und readExternal implementieren, um die Serialisierung und Deserialisierung der Objekte zu steuern.
- **Wie kann man einzelne Attribute nicht serialisieren?**
    - In Java gibt es zwei Möglichkeiten, um einzelne Attribute nicht zu serialisieren:
        - Das "transient" Schlüsselwort kann verwendet werden, um ein Feld als nicht serialisierbar zu markieren.
        - Sie können die writeObject() und readObject() Methoden implementieren, um die Serialisierung und Deserialisierung von Objekten manuell zu steuern. In diesen Methoden können Sie festlegen, welche Felder serialisiert werden sollen und welche nicht.
- **Wie wird serialVersionUID generiert?**
    - Ähnlich wie eine Hashfunktion werden alle Parameter betrachtet und daraus die UID generiert → eine Änderung der Parameter/Attribute einer Klasse führt zu einer anderen UID
    - Die serialVersionUID in Java wird automatisch generiert, wenn eine Klasse das Serializable-Interface implementiert und keine eigene serialVersionUID definiert. Die generierte serialVersionUID basiert auf verschiedenen Aspekten der Klasse, wie z.B. dem Namen der Klasse, der Namen und der Signatur der verwendeten Felder und Methoden. Wenn eine eigene serialVersionUID definiert wird, wird diese verwendet, um die Kompatibilität zwischen verschiedenen Versionen der Klasse sicherzustellen.
- **Was ist der Unterschied zwischen Swing und AWT?**
    - Swing und AWT sind beide GUI-Toolkits für Java, aber Swing bietet einen plattformunabhängigen Look and Feel, eine größere Auswahl an Komponenten und ist flexibler und erweiterbar. AWT hingegen ist schneller, aber weniger flexibel und bietet eine begrenztere Auswahl an Komponenten.

- **Was sind Special Purpose Container und wofür sind Sie nützlich? || Das selbe für den General Purpose Container**
  - In Java gibt es zwei Arten von Containern: Special Purpose Container und General Purpose Container.
  - Special Purpose Container: Special Purpose Container sind Container, die für einen bestimmten Zweck oder eine bestimmte Aufgabe entwickelt wurden. Ein Beispiel für einen Special Purpose Container ist das JFrame in Swing, das speziell für die Erstellung von Fenstern in einer GUI-Anwendung entwickelt wurde. Ein weiteres Beispiel ist das JApplet, das für die Entwicklung von Applets verwendet wird.
  - Special Purpose Container sind nützlich, da sie spezielle Funktionen und Eigenschaften bieten, die für bestimmte Anwendungsfälle benötigt werden. Sie bieten eine einfachere Möglichkeit, bestimmte Aufgaben auszuführen, ohne dass der Entwickler den Container selbst erstellen muss.
  - General Purpose Container: General Purpose Container sind Container, die für eine Vielzahl von Anwendungsfällen verwendet werden können. Ein Beispiel für einen General Purpose Container ist das JPanel in Swing, das für die Erstellung von allgemeinen Komponenten in einer GUI-Anwendung verwendet wird.
  - General Purpose Container sind nützlich, da sie eine breitere Palette von Anwendungsfällen abdecken und dem Entwickler mehr Freiheit bei der Gestaltung und Erstellung von GUI-Anwendungen geben. Sie können für die Erstellung von allgemeinen Layouts, Schaltflächen, Textfeldern und anderen Komponenten verwendet werden.
- **Nennen Sie die zwei Mechanismen, die Ihnen im Rahmen von Swing zur Verfügung stehen, um eigene Klassen zur Behandlung von Events zu implementieren. Beschreiben Sie außerdem kurz den grundlegenden Unterschied.**
  - Im Rahmen von Swing stehen Ihnen das Listener-Modell und das Adapter-Modell zur Verfügung, um eigene Klassen zur Behandlung von Events zu implementieren. Der Unterschied besteht darin, dass das Listener-Modell mehr Flexibilität bietet, während das Adapter-Modell einfacher zu implementieren ist
- **Nennen Sie die drei Akteuere des MVC-Pattern? Beschreiben Sie kurz deren jeweilige Verantwortlichkeit.**
  - Die drei Akteure des MVC-Pattern sind:
    - Model: Das Model repräsentiert die Daten und die Geschäftslogik der Anwendung. Es ist für die Verwaltung der Daten und die Durchführung von Berechnungen und Operationen zuständig.
    - View: Die View ist für die Darstellung der Daten und die Interaktion mit dem Benutzer verantwortlich. Sie erhält Daten vom Model und stellt sie dem Benutzer auf eine geeignete Weise dar.
    - Controller: Der Controller ist für die Koordination zwischen Model und View zuständig. Er empfängt Benutzereingaben von der View und leitet sie an das Model weiter, um die Daten zu aktualisieren. Er empfängt auch Benachrichtigungen vom Model über Änderungen an den Daten und aktualisiert dann die View entsprechend.
  - Das MVC-Pattern trennt die Verantwortlichkeiten klar zwischen den drei Akteuren, um eine klare Struktur und eine bessere Wartbarkeit der Anwendung zu erreichen.
- **Was sind anonyme Klassen?**
  - Anonyme Klassen in Java sind Klassen, die direkt bei der Verwendung definiert und instantiiert werden können, ohne sie vorher als eigene Klasse zu definieren. Sie werden oft verwendet, um eine Implementierung eines Interfaces oder einer abstrakten Klasse direkt an Ort und Stelle zu definieren und zu instantiieren.

In Java gibt es einen Unterschied zwischen dem Instanziieren und dem Initialisieren von Variablen oder Objekten.
Instanziieren bezieht sich auf die Erstellung eines Objekts einer Klasse. Wenn eine Klasse in Java definiert wird, ist sie nur ein Bauplan oder eine Schablone, die beschreibt, welche Eigenschaften und Methoden ein Objekt haben sollte. Wenn ein Objekt dieser Klasse erstellt wird, wird es instanziiert. Das bedeutet, dass es eine tatsächliche Kopie der Klasse erstellt, die alle Eigenschaften und Methoden der Klasse enthält. Dies geschieht normalerweise mit dem "new" -Schlüsselwort, gefolgt vom Namen der Klasse.
Initialisieren bezieht sich auf die Festlegung des Anfangswerts einer Variablen oder Eigenschaft eines Objekts. Wenn eine Variable oder ein Objekt instanziiert wird, hat sie in der Regel einen Standardwert, der je nach Datentyp unterschiedlich ist. Wenn Sie jedoch einen spezifischen Anfangswert festlegen möchten, können Sie dies durch Initialisierung tun. In Java können Sie eine Variable oder ein Objekt während der Deklaration initialisieren, indem Sie ihr einen Wert zuweisen, oder Sie können sie später im Code initialisieren, indem Sie ihr einen Wert zuweisen.
Zusammenfassend lässt sich sagen, dass Instanziieren die Erstellung eines Objekts einer Klasse darstellt, während Initialisieren die Festlegung des Anfangswerts einer Variable oder Eigenschaft eines Objekts ist.




Swing Components in Java
Swing Components
ImageIcon
JButton
JLabel
JPasswordField
Jtext
JCheckBox
JRadioButton
JList
JComboBox
JFileChooser
JTextField
JTextArea
www.educba.com