|  |  |  |  |
| --- | --- | --- | --- |
|  | Konkrete Klasse | Abstrakte Klasse | Interface |
| Definiert durch | public class … { | public **abstract** class … { | public **interface** … { |
| Kann erben von | Maximal einer konkreten Klasse 🡪 **extends** | Maximal einer konkreten Klasse 🡪 **extends**  Implementierte Methoden können wieder auf abstract gesetzt werden | Kann **nicht** von konkreten Klassen erben |
| **Oder** maximal einer abstrakten Klasse 🡪 **extends** Subklasse muss alle abstrakten Methoden implementieren | **Oder** maximal einer abstrakten Klasse 🡪 **extends**  Implementierte Methoden können wieder auf abstract gesetzt werden, abstrakte Methoden können (aber müssen nicht) implementiert werden | Kann **nicht** von abstrakten Klassen erben |
| **Zusätzlich** mehreren Interfaces 🡪 **implements**  Subklasse muss alle abstrakten Methoden implementieren oder erben  Mehrfache default-Methoden müssen überschrieben werden | **Zusätzlich** mehreren Interfaces 🡪 **implements**  Implementierte **static**-Methoden des Super-Interfaces können **nicht**, aber implementierte **default**-Methoden können schon wieder auf **abstract** gesetzt werden  Abstrakte Methoden des Super-Interfaces können (aber müssen nicht) implementiert werden  Mehrfach vorhandene default-Methoden der Super-Interfaces mit der selben Signatur müssen überschrieben werden | **Mehreren** Interfaces 🡪 **extends**  Implementierte statische oder default-Methoden des Super-Interfaces können auf abstrakt gesetzt werden  Abstrakte Methoden des Super-Interfaces können (aber müssen nicht) implementiert werden, aber wenn, dann nur als **static** oder **default** |
| Konstruktor | Kann mehrere mit unterschiedlichen Signaturen haben  Wenn kein parameterloser existiert, erstellt Java selbst einen  Die Klasse kann instanziert werden | Kann mehrere mit unterschiedlichen Signaturen haben, diese können aber **nur in Konstruktoren von Subklassen** aufgerufen werden  Wenn keiner existiert, erstellt Java **keinen**  Die Klasse kann **nicht** direkt instanziert werden (nur über nicht abstrakte Subklassen) | **Keine** Konstruktoren |
| Methoden | Kann **Klassen-** (**static**) und **Objektmethoden** haben  Kann **final**-Methoden haben  Alle müssen implementiert sein (**keine abstrakten** Methoden)  Alle access modifier erlaubt | Kann, aber muss nicht, implementierte Methoden haben: Alle access modifier erlaubt, können **Klassen-** (**static**) oder **Objektmethoden** sein, können **final** sein  Kann, aber **muss nicht**, **abstrakte** Methoden haben: Alle access modifier **außer private** erlaubt, aber nur **Objektmethoden** und **nicht final** | **Alle Methoden** sind **immer automatisch public**, **nie final** und eine der drei Arten:  **default**-Methoden: Sind **Objektmethoden**, haben eine Implementierung  **static**-Methoden: Sind **Klassenmethoden**, haben eine Implementierung  **Abstrakte** Methoden: Sind **Objektmethoden**, haben **keine Implementierung** |
| Attribute | Alle access modifier möglich  **static** und **nicht static** möglich  **final** möglich | Alle access modifier möglich  **static** und **nicht static** möglich  **final** möglich | **Immer automatisch public, static und final** |