Java Code Conventions:

Benennungsregeln:

- Variablen-, Parameter- und Methodenbezeichner beginnen mit kleinem Buchstaben.
- Klassen, Interface- und Enumbezeichner beginnen mit großem Buchstaben.
- Werden mehrere Wörter aneinandergehangen, dann wird die CamelCase-Notation verwendet.
- Packages müssen durchgehend klein geschrieben werden.
- Die Benamung erfolgt auf Englisch.
- Konstanten werden grundsätzlich nur mit Großbuchstaben benannt und die einzelnen Wörter werden durch einen Unterstrich getrennt. (bspw. MAX_SIZE)
- Bezeichner für Methoden sind im Imperativ zu wählen: handleInvalidInput(), findBraces(), usw. – NICHT invalidInput(), braces()
- Methoden die einen boolean zurück liefern oder Variablen vom Typ boolean beginnen mit is, bspw. isInputValid oder isUserActive() – Gelegentlich macht es auch Sinn has, can oder should zu verwenden – hasLicense(), canEvaluate(), shouldAbort(), ...
- Genericbezeichner werden grundsätzlich mit nur einem Großbuchstaben benannt, z.B. T
- Für Bezeichner wird immer ein selbstsprechender Name gewählt werden (Ausnahme: siehe Kommentare).
    - Hierzu zählt auch die zweckgemäße Klassen Benamung wie z.B. das Utility-Klassennamen mit dem Wort Utils oder mindestens mit einem "s" (plural) enden (bspw. StringUtils oder Arrays).

Formatierungsregeln:

- Die geöffnete geschweifte Klammer steht in der gleichen Zeile wie die Methodensignatur, die Verzweigung oder die Schleife.
- Hinter Verzweigung- und Schleifenschlüsselwörtern folgt vor den Klammern eine Leerstelle.
- Zwischen geschweifter Klammer und runden Klammern bzw. Java-Schlüsselwörtern steht ein Leerzeichen. Vor und nach geschweiften Klammern steht ein Leerzeichen.
- Nach und vor jedem arithmetischen, Vergleichs, Zuweisungs oder booleschen Operator wird ein Leerzeichen gesetzt.
- Die zusätzliche Einrückungstiefe nach jeder geöffneten geschweiften Klammer oder den Cases bei einer Switch-Anweisung beträgt vier Leerzeichen.
- Die zusätzliche Einrückungstiefe bei einer umgebrochenen Zeile beträgt acht Leerzeichen.
- Ein Tabulatorzeichen zählt nicht als Leerzeichen.

Zugriffsmodifikatoren und Instanzen:

- Auf Instanzvariablen wird *immer* einheitlich über **this** zugegriffen.
- Der Zugriff über **this** bei Methodenaufrufen ist optional
- Standardmäßig haben alle Methoden und Variablen als Zugriffsmodifikator private.
- Die Benutzung der default-Sichtbarkeit wird vermieden oder kommentiert.
- Das Schlüsselwort **static** wird nur sinnvoll und nicht aus Bequemlichkeit verwendet.

Blockregeln:

- Bei Verzweigungen und Schleifen wird immer ein Block angeschlossen, auch wenn nur eine einzelne Anweisung folgt.
- Blöcke die nicht aus Kontrollstrukturen, Methoden, Initialisierungsblöcken, o.ä. resultieren, werden vermieden.
- Blöcke beinhalten nicht mehr als 60 Zeilen (dazu zählen auch Methodenblöcke).
- Ressourcen, wie z.B. eine Datenbankverbindung, müssen nach der Benutzung wieder geschlossen werden.

Kommentare:

- Methoden/Klassen/Variablen/o.ä., die nicht eindeutig benannt werden können, müssen mit einem JavaDoc-Kommentar versehen werden.
- Jede Klasse muss mit einem @author-Kommentar (im Format: Vorname Nachname) versehen werden.
- Inline-Kommentare sollten nach Möglichkeit vermieden werden.
- Auskommentierter Code muss begründet oder entfernt werden.

Klassenstruktur:

- Attribute einer Klasse werden ganz oben deklariert werden.
- Danach folgen die Konstruktoren und dann erst die Methoden der Klasse.
- Optional: Zusammenhängende Methoden sollten nach Möglichkeiten untereinander geschrieben werden.
- Optional: Methoden die von Object überschrieben werden (toString, equals, o.ä.) sollten sich ganz unten in der Klasse befinden.
- Logikabschnitte (package/import/Klasse/Methode/etc.) werden durch eine Leerzeile getrennt werden.
- Variablen werden immer so lokal wie möglich deklariert werden.
- Unbenutzte Variablen und Imports sowie Überflüssige vergleiche (bspw. if(boolscheVariable == true)) müssen entfernt werden.
- In einem fertigen Programm dürfen nur für die Funktionalität der Anwendung

notwendige Konsolenausgaben enthalten sein.
- Sich wiederholende Logik sollte nach Möglichkeit in eigene Methoden ausgelagert werden.
- In der Klasse in der die main-Methode steht werden nur allgemeingültigen Methoden, wie z.B. start oder initDatabase, definiert.
- Neue Ansichten in einer Anwendung werden in neuen Klassen definiert, damit Inhalte von Klassen möglichst genau definiert werden können.
- Hilfsklassen und Enums haben immer einen privaten Konstruktor.

Ordnerstruktur:

- Bei größeren Projekten (> 4 Dateien) wird das Projekt in Packages unterteilt werden.
- Optional: Ressourcen, wie z.B. Bilder oder Datenbankdateien, sollten in einen Ressourcenordner ausgelagert werden.
- Optional: Packages sollten nach Möglichkeit mit der Domäne der Firma benannt werden. (Bsp.: de.volkswagen.mypackage für das package mypackage das von einem Volkswagenmitarbeiter erstellt wurde.)

Tests:

- Der Klassenname ist eine Kombination aus der zu testenden Klasse/Funktionalität und dem Wort "Test".
- Der Testmethodenname ist eine Kombination aus dem Wort "test" und der zu testenden Funktionalität sein
- Testmethoden werden immer mit einem Javadoc Kommentar beschrieben. In diesem wird die grundsätzliche Funktionalität des Tests beschrieben werden.

Git:

- Commitnachrichten müssen stets aussagekräftig und auf englisch sein.
- Es werden nur sinnvolle Zwischenergebnisse ins repository geladen und nicht "Ich habe ein Leerzeichen ergänzt"
- Branches werden vorne immer nach ihrem Verwendungszweck benannt (bugfix/feature/etc) dann ein "/", dahinter die Story/die Beschreibung der Tätigkeit mit Bindestrich (bspw. "feature/Create-AdminView").