**Zusammenfassung – Review – Klassen *Article* *Person***

Mitschrift für? Word • Stand: 11.09.2025

**1. Kontext**

Im Projekt wurden zwei C #-Klassen analysiert: **Article** und **Person**. Ziel war eine saubere Datenstruktur mit klarer Validierung, konsistenter Benennung und stabiler Ausgabe.

**2. Kernprobleme (kurz und ehrlich)**

**• Doppelte Datenquelle / Inkonsistenz:** Im *Article*-Konstruktor werden Felder gesetzt, *ToString()* liest jedoch Auto-Properties → angezeigte Werte können abweichen.

**• Tipp-/Benennungsfehler:** *Articlename* vs. *Arcticlename*; in *Person* *Firstname/Lastname* vs. *FirstName/LastName* in *ToString()*.

**• Validierung schwach:** *Id*, *Price*, *StockQuantity* korrigieren teils still (ignoriert/auf 0) statt klar zu scheitern.

**• Bool-Redundanz:** *IsAvailable* wird gesetzt, obwohl es meist aus dem Bestand ableitbar ist.

**• Konventionen:** Properties/Methoden sollten *PascalCase*, Felder *‑camelCase* sein.

**• Ausgabe:** Preis ohne W*ToString()*.

**3. Empfehlungen (Soll-Zustand)**

**• Eine Quelle f** Ctor setzt konsequent die *Properties* (Validation greift); nicht Felder und Autos mischen.

**• Strenge Validation / Fail-fast:** Bei < 0 f*Id*, *Price*, *StockQuantity* *ArgumentOutOfRangeException* werfen; Strings auf Leerheit pr*Trim()*.

**• Abgeleitete Verf** public bool IsAvailable => StockQuantity > 0;

**• Konsistente Namen:** *ArticleName*, *StockQuantity*, *Category*; in *Person* *Firstname/Lastname* konsistent zu *ToString()* verwenden.

**• Saubere Ausgabe:** Preis mit W

**• Klare API:** Domain-Methoden wie *Rename*, *Reprice*, *AdjustStock(int delta)* mit Validierung.

**4. Quick Wins (heute fixbar)**

**\_0** Schreibweisen fixen (*ArticleName/Category/StockQuantity*; *Firstname/Lastname*).

**\_0** Konstruktor auf Property-Zuweisungen umstellen (nicht direkt Felder setzen).

**\_0** *IsAvailable* aus *StockQuantity* ableiten; Setter entfernen.

**\_0** *ToString()* korrigieren (richtige Properties, Price:C).

**5. Warum sich das lohnt**

Weniger stille Datenfehler, fr

*Kurzfazit:* **Saubere Properties, harte Validation, konsistente Namen** ࢐ **stabile Klassen.**