

# Aufgabenblatt: Flexbox in CSS

## Teil 1: Grundlagen verstehen

Beantworte die folgenden Fragen schriftlich:

- Was ist der Unterschied zwischen einem normalen Block-Layout und einem Flex-Layout?
- Welche Elemente werden zu Flex-Items?
- Welche Eigenschaft aktiviert Flexbox auf einem Container?

## Teil 2: Achsen verstehen

Gegeben ist folgendes CSS:

```
.container {  
  display: flex;  
  flex-direction: row;  
}
```

- In welche Richtung verläuft die Hauptachse?
- Wo liegt in diesem Fall die Querachse?

Ändere anschliessend `flex-direction` auf `column` und beschreibe die Unterschiede.

## Teil 3: Ausrichtung im Container

Erstelle ein eigenes Beispiel mit mindestens drei Boxen und einem Flex-Container.

Teste nacheinander:

- `justify-content: center;`
- `justify-content: space-between;`
- `align-items: center;`

Dokumentiere jeweils kurz, was sich sichtbar verändert.

## Teil 4: Umbruch und Abstände

Gegeben ist folgendes CSS:

```
.container {  
  display: flex;  
  flex-wrap: wrap;  
  gap: 12px;  
}
```

- Was passiert, wenn der Platz in einer Zeile nicht mehr ausreicht?
- Welchen Vorteil hat `gap` gegenüber Margins auf den einzelnen Items?

Erstelle ein Beispiel mit mindestens sechs Elementen, sodass ein Zeilenumbruch sichtbar wird.

## Teil 5: Eigenschaften von Flex-Items

Untersuche folgende Werte:

```
.item-a { flex: 1; }  
.item-b { flex: 2; }  
.item-c { flex: 1; }
```

- Welches Element bekommt am meisten Platz?
- Wie verändert sich das Layout, wenn alle Elemente `flex: 1`; erhalten?

## Teil 6: Fehler finden und korrigieren

Der folgende Code enthält konzeptionelle Fehler:

```
.item {  
  justify-content: center;  
  align-items: center;  
}
```

- Warum funktionieren diese Eigenschaften hier nicht wie erwartet?
- Korrigiere den Code so, dass die Ausrichtung korrekt funktioniert.

## Teil 7: Praxisaufgabe

Erstelle eine kleine Startseite mit:

- einer Navigationsleiste mit Logo und drei Links
- einem Kartenbereich mit mindestens vier Karten
- einem Footer

Verwende Flexbox für alle drei Bereiche und achte auf:

- saubere Struktur
- sinnvolle Klassennamen
- lesbaren und eingerückten Code

## **Teil 8: Reflexion**

Beantworte die folgenden Fragen:

- Welche Flexbox-Eigenschaft war für dich am hilfreichsten?
- Bei welcher Aufgabe hattest du Schwierigkeiten und warum?
- In welcher realen Webseite könntest du Flexbox direkt einsetzen?