0.1. Übung 1

(a). HTML

Tags Die Auszeichnung von Seiteninhalten wird in HTML durch Tags gesteuert. Mit Ausnahme weniger spezieller Tags (etwa `<br>`) werden stets öffnende und schließende Tags verwendet um den auszuzeichnenden Inhalt zu strukturieren:

```
<html>
  <head>
    <title>Seitentitel</title>
  </head>

  <body>
    <h1>Eine Überschrift</h1>
  </body>
</html>
```

`head` Das `head`-Tag beinhaltet Informationen über den in `body` folgenden Inhalt und Referenzen zu weiteren anzeigerelevanten Dokumenten. Es darf nur nur einmal im Dokument vorkommen, steht direkt zu Beginn und muss ein `title`-Element beinhalten. Darüber hinaus sollte in `head` die Zeichenkodierung des Dokuments definiert werden.

```
<html lang="de">
  <head>
    <meta charset="utf-8">
    <link rel="stylesheet" type="text/css" href="styles.css"
       media="all">
    <title>Ein Seitentitel</title>
  </head>
  <body>
    [..]
  </body>
</html>
```

`img` Das `img`-Tag bindet Bilddaten in die Webseite ein. `img` ist, wie `br`, ein Standalone-Tag ohne Inhalt und ohne schließendes Tag. Zwingend erforderlich sind jedoch die Attribute `src` und `alt`, mit denen die Bildquelle und ein Ersatztext definiert werden. Der Ersatztext wird angezeigt, falls das Bild nicht geladen oder dargestellt werden kann.

```
<img src="./katze.png" alt="Eine Katze">
```

`table` Tabellarische Inhalte auf Webseiten können mit dem `table`-Tag dargestellt werden. Eine minimale Tabellenstruktur kann mithilfe der *table row*, *table head*- und *table data cell*-Tags (`tr, th, td`) wie folgt aussehen:

```
<table>
  <tr>
    <th>Überschrift Spalte 1</th>
    <th>Überschrift Spalte 2</th>
  </tr>
  <tr>
    <td>Zelle 1</td>
    <td>Zelle 2</td>
  </tr>
  <tr>
    <td>Zelle 3</td>
    <td>Zelle 4</td>
  </tr>
</table>
```

Es empfiehlt sich, Tabellen in einen `thead`, mindestens einen `tbody` und optional einen `tfooter` aufzuteilen. So ist es beispielsweise möglich, mittels CSS die Kopfzeile einer Tabelle im Browser zu fixieren, während der Benutzer durch längere Tabellen scrollt.

(b). CSS

Syntax Die grundlegende Syntax von Cascading Stylesheets (CSS) sieht wie folgt aus:

```
Selektor { Eigenschaft: Wert; }
```

Es ist möglich, mit verschiedenen Selektoren demselben Element mehrfach Stileigenschaften zuzuweisen. Wird in einem Stylesheet mehrmals dieselbe Eigenschaft auf demselben Element verändert, so wird die zuletzt verarbeitete Anweisung verarbeitet und dargestellt.

Stil-Eigenschaften Die verschiedenen Anzeigeelemente haben individuelle Sätze von Eigenschaften, die in Stylesheets verändert werden können. Es folgen einige Beispiele:

```
h1 { color: red; }
div {
    width: 260px;
    margin: auto;
}
p {
    color: white;
    font-size: large;
```

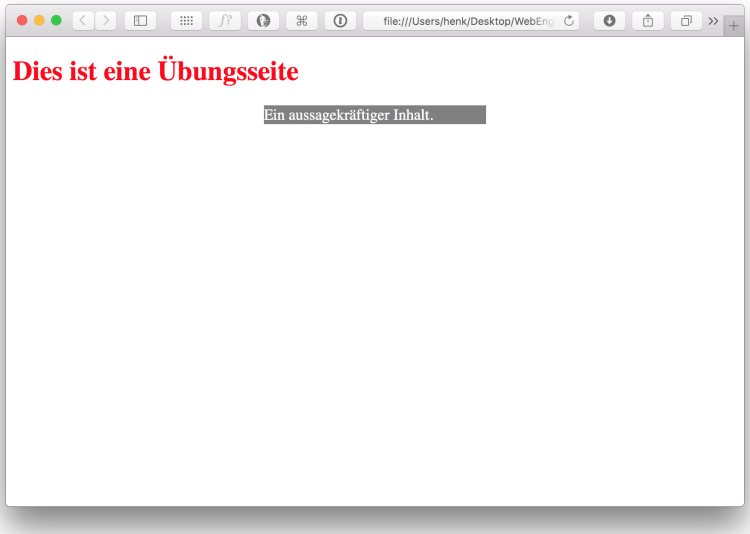

Abbildung 0.1.: Darstellung des beispielhaften CSS

```
9       background-color: gray;
10  }
```

Einbettung in HTML Es gibt verschiedene Möglichkeiten CSS in Webseiten einzubinden:

`link`
Beispiel:

```
1  <link rel="stylesheet" href="stylesheet.css">
```

Mithilfe des `link`-Tags können ein oder mehrere extern definierte Stylesheets in eine Webseite integriert werden. Dies erlaubt die höchste Wiederverwendbarkeit und sorgt für eine übersichtliche Trennung von Auszeichnung (HTML) und Darstellung.

zentrales `style`-Element
Beispiel:

```
1  <style>
2    h1 { color: green; }
3  </style>
```

Das zentrale style-Element erlaubt es, direkt im HTML-Dokument Stilanpassungen vorzunehmen.

`style`-Attribut für einzelne Elemente

Beispiel:

```
<h1 style="color: green;">
  Dies ist eine Überschrift
</h1>
```

Viele HTML-Elemente besitzen ein `style`-Attribut, das es erlaubt, Stilanpassungen direkt am Element vorzunehmen. Diese *Inline-Styles* sind jedoch selten von Vorteil, da sie für jedes neue Element wieder definiert werden müssten und so den Wartungsaufwand erheblich erhöhen.

Selektoren/Kombinatoren Selektoren dienen im Wesentlichen dazu, einer bestimmten Menge von Elementen Stileigenschaften zuzuweisen. Um einen möglichst präzisen Zugriff auf bestimmte Elemente zu erlauben, gibt es eine Vielfalt an verschiedenen Selektoren.

Typselektoren

Der Typselektor wählt alle Elemente eines Typs aus:

```
<style>
  p { color: red; }
</style>

<div>
  <p>
    Ein Absatz <!-- rot -->
  </p>
</div>

<p>
  Ein weiterer Absatz <!-- rot -->
</p>
```

Klassenselektor `.` / ID-Selektor `#`

Zur Auswahl aller Elemente denen mit dem `class`- oder `id`-Attribut eine bestimmte Klasse/ID zugewiesen wurde.

```
<style>
  .content-pane { background-color: red; } /*
     Klassenselektor */
  #special { background-color: blue; } /* ID-Selektor */
</style>
<!-- rot -->
<div class="content-pane">
  ...
</div>
```

```
<!-- blau -->
<div id="special">
  ...
</div>
<!-- neutral -->
<div>
  ...
</div>
```

Kindselektor `a > b`

Wählt ein Element `b` nur dann aus, wenn es ein Kindelement von `a` ist.

```
<style>
  div > p { color: red; } /* färbt den Text aller
                             p-Elemente innerhalb
                             eines div-Elements rot */
</style>

<div>
  <p>Ein Absatz</p> <!-- rot -->
</div>

<p>Ein weiterer Absatz</p> <!-- neutral -->
```

Nachfahrenselektor `a b`

Ähnlich zum Kindselektor, wählt jedoch auch weiter entfernte Nachfahren.

```
<style>
  div a { font-weight: bold; } /* Wählt ein Element a
                                  auch dann aus, wenn
                                  es in einem anderen
                                  Element enthalten
                                  ist, solange es ein
                                  Nachfahre eines
                                  div-Elements ist. */
</style>

<div>
  <!-- Der Link wird fett dargestellt -->
  <a href="_blank">Ein Link</a>
  <p>
    <!-- Dieser Link wird ebenfalls fett dargestellt -->
    <a href="_blank">Noch ein Link</a>
  </p>
</div>
```

Universalselektor `*`

Der Universalselektor selektiert zunächst alle Elemente:

```
<style>
  * { color: red; }
</style>

<h1>Eine Überschrift</h1> <!-- rot -->
<p>Ein Absatz.</p> <!-- rot -->
```

Jedoch kann der Universalselektor auch mit anderen Selektoren kombiniert werden, z.B. dem Nachfahrenselektor:

```
<style>
  div * { color: red; }
</style>

<p>Ein Absatz</p> <!-- nicht rot -->
<div>
  <h2>Eine Überschrift</h2> <!-- rot -->
  <p>Ein Absatz</p> <!-- rot -->
</div>
```

Nachbarselektor `a + b`

Wählt Element `b` nur dann aus, wenn es direkter nachfolgender Nachbar von `a` ist.

```
<style>
  h1 + p { font-weight: bold }
  p + p { font-style: italic }
</style>

<h1>Der Nachbarselektor</h1>
<p>Erster Absatz.</p> <!-- bold -->
<p>Zweiter Absatz.</p> <!-- italic -->
<p>Dritter Absatz.</p> <!-- italic -->
<div>neutrales Element</div>
<p>Vierter Absatz</p> <!-- neutral -->
```

Geschwisterselektor `a ~ b`

Selektiert alle Elemente `b`, die auf derselben Ebene wie `a` sind und auf `a` folgen.

```
<style>
  h1 ~ p {font-weight: bold;}
</style>

<p>Erster Absatz.</p>
<h1>Der Geschwisterselektor</h1>
<p>Zweiter Absatz.</p> <!-- bold -->
<hr>
<p>Dritter Absatz.</p> <!-- bold -->
```

(c). Aufgaben

Aufgabe (a)
Erstellen Sie eine HTML-Webseite mit dem Titel “Übungstest”, einer Überschrift “Dies ist eine Übungsseite” und einer Tabelle mit drei Zeilen und zwei Spalten. Die erste Tabellenspalte soll jeweils ein Bild enthalten und die zweite Spalte die dazugehörige Beschreibung. Verwenden Sie in der ersten Zeile der Tabelle den `tablehead`-Tag um eine Beschreibung der Spalten hinzuzufügen. Beschränken Sie die Breite der Bilder in der Anzeige auf 120 Pixel.

Aufgabe (b)
Setzen Sie für die Bilder der Tabelle Links, sodass ein Klick auf das Bild ein neues Browser-Fenster öffnet. Die Links sollen jeweils zur Bildquelle führen.

Aufgabe (c)
Da Sie nicht weiterhin für jedes einzelne Bild innerhalb der ersten Tabellenspalte eine Breite definieren wollen, sollen Sie nun die Breite für alle Bilder innerhalb der ersten Tabellenspalte begrenzen. Verwenden Sie dazu in das HTML eingebettetes CSS.

Aufgabe (d)
Für eine bessere Lesbarkeit der Tabelle möchten Sie nun eine alternierende Färbung der Tabellenzeilen einführen. Jede zweite Tabellenzeile soll die Hintergrundfarbe `#ccc` bekommen. Verwenden Sie dazu in das HTML eingebettetes CSS.

Aufgabe (e)
Stellen Sie die Tabelle nun mithilfe von CSS mittig im Browserfenster dar.

Aufgabe (f)
Der Hintergrund einer Tabellenzeile soll mithilfe von CSS die Farbe `#aaa` bekommen, sobald der Mauszeiger darüber liegt.