CSS3
Cascading Style Sheets Level 3

Grundlagen

(Stand 2024)

CSS3_2024

Isolde Kommer, Marc Haunschild

1. Ausgabe, Januar 2024

ISBN 978-3-98569-177-7

Fortgeschrittene CSS-Techniken

Weiterführende CSS-Themen

Bevor Sie beginnen …

Voraussetzungen und Ziele

Hinweise zu Soft- und Hardware

Nachdem CSS als Standard verabschiedet wurde, begann die Implementierung in den Browsern. Inzwischen ist diese in allen aktuellen Browsern sehr weit fortgeschritten. Um fehlerhafte Darstellungen von Webseiten zu vermeiden, sollten Sie daher aktuelle Browserversionen verwenden. Generell lässt sich sagen: Je älter die verwendete Browserversion, desto schwächer ist die Unterstützung von Webstandards (das betrifft nicht nur CSS, sondern zum Beispiel auch HTML und JavaScript).

Sofern nicht anders vermerkt, wird im Folgenden davon ausgegangen, dass Sie die unten genannten Browser oder neuere Versionen verwenden.

	Name	Erhältlich unter der Internetadresse:
	Microsoft Edge	*https://www.microsoft.com/de-de/windows/microsoft-edge*
	Firefox	*https://www.mozilla.org/de/firefox/*
	Opera	*https://www.opera.com/de*
	Safari	*https://www.apple.com/de/safari/*
	Chrome	*https://www.google.com/chrome/browser/desktop/index.html*

Die meisten Anwender verwenden relativ neue Versionen. Das liegt nicht zuletzt an den immer komfortableren Update-Mechanismen der Browser – bei Google Chrome bekommt der Anwender davon gar nichts mehr mit. Dennoch gilt das nicht für jeden Besucher Ihrer Website. Daher ist es sinnvoll, sich die eigene Seite auch mal mit älteren Browsern anzusehen.

Konventionen

Hervorhebungen im Text

Damit Sie bestimmte Elemente auf einen Blick erkennen und zuordnen können, werden diese im Text durch eine besondere Formatierung hervorgehoben.

Kursivschrift	Bezeichnungen für Programmelemente wie Register etc. sowie Dateinamen, Ordnernamen und Internetadressen werden *kursiv* geschrieben.
Fettschrift	Wichtige Begriffe werden **fett** hervorgehoben.

`Courier new`	Programmtext, Programmnamen, Funktionsnamen, Variablennamen, Datentypen, Zeichen, Operatoren etc.
`Courier New Kursiv`	kennzeichnet Zeichenfolgen, die vom Anwendungsprogramm ausgegeben oder ins Programm eingegeben werden.

[ ] Bei Darstellungen der Syntax einer Programmiersprache kennzeichnen eckige Klammern optionale Angaben.

/ Bei Darstellungen der Syntax einer Programmiersprache werden alternative Elemente durch einen Schrägstrich voneinander getrennt.

In den Beispielen wird teilweise ein Blindtext verwendet, der nur dem Anschein nach in lateinischer Sprache verfasst ist. Dieser Blindtext entspricht im Satzbild (Länge der Wörter und Sätze) durchschnittlichem deutschen Mengentext. Der Text ergibt inhaltlich keinen Sinn und dient lediglich als Beispiel.

HERDT BuchPlus – unser Konzept:

Problemlos einsteigen – Effizient lernen – Zielgerichtet nachschlagen

(weitere Infos unter *www.herdt.com/BuchPlus*)

Nutzen Sie dabei unsere maßgeschneiderten, im Internet frei verfügbaren Medien:

1

Verwendung von CSS

1.1 Warum Formatvorlagen verwenden?

In Apps zur Textverarbeitung wie etwa Microsoft Word oder Open Office können Sie Text direkt formatieren (zum Beispiel eine Überschrift markieren und eine größere Schrift verwenden, was sich nur auf den ausgewählten Text auswirkt) oder die Vorgaben für alle Überschriften anpassen. Diese werden in Formatvorlagen zusammen mit dem Aussehen für alle anderen Texte (z. B. Listen und Absätze) gespeichert. Wenn Sie Formate in einer Vorlage festlegen (zum Beispiel größere Schrift für die Überschrift erster Ordnung), werden alle Überschriften erster Ordnung größer. Durch die Verwendung von Formatvorlagen wird sichergestellt, dass gleichartige Texte identisch aussehen.

Ganz ähnlich funktionieren Cascading Style Sheets (CSS): Sie schreiben CSS-Formatvorlagen für Textauszeichnungssprachen wie HTML oder XML. In diesem Buch wird auf die Formatierung von Webseiten, also HTML-Dokumenten, eingegangen.

Mit Stylesheets formatierte Webseite (kap01\start.htm)

In diesem Beispiel (*kap01\start.htm*) wurden der Überschrift, den beiden Absätzen und dem Text im Kasten per CSS Formate zugewiesen.

Webseiten können und sollten ausschließlich mit CSS formatiert werden.

1.2 Vorteile von CSS

- **Trennung von Inhalt und Präsentation**
  Formatvorlagen (Stylesheets) dienen der Gestaltung logisch strukturierter Dokumente wie HTML oder XML. Wird die Trennung von Inhalt (HTML) und Präsentation (CSS) konsequent eingehalten, kann die Darstellung von Webseiten im Browser weitgehend beeinflusst werden, ohne die Inhalte zu manipulieren.
- **Öffentliche Dokumentation**
  Die Standards, die CSS zugrunde liegen, sind öffentlich dokumentiert. Alle Browserhersteller können somit diese Standards unterstützen. Das zugrunde liegende Betriebssystem spielt keine Rolle mehr, wodurch Webseiten im Idealfall identisch aussehen – egal ob Windows, Linux oder macOS verwendet wird.
- **Unterstützung unterschiedlicher Medien**
  CSS erlaubt es, Formatierungsanweisungen ganz gezielt für bestimmte Geräte anzulegen. Das bedeutet, dass die Inhalte zum Beispiel für den Druck anders aufbereitet werden können, als sie auf einem Bildschirm angezeigt werden.
- **Unterstützung verschieden großer Bildschirme**
  Sie können unterschiedlich große Bildschirme gezielt ansprechen. Dadurch können Sie für Smartphones, Tablets, Laptops und ultrahochauflösende Bildschirme angepasste Layouts ausliefern.
- **Effizienz**
  Bei konsequenter Anwendung wird ein einheitliches Layout sichergestellt. Änderungen, die sich auf alle einzelnen Seiten auswirken, müssen nur an einer Stelle vorgenommen werden.
- **Verständlichkeit**
  CSS ist eine modulare Sprache, die für Menschen gut lesbar ist.

Durch die stetige Weiterentwicklung kommt es dazu, dass nicht alle Browser gleichzeitig alle Neuerungen korrekt umsetzen. Das kann vor allem bei der Verwendung von neuen CSS3-Features zu Unterschieden in den Browser-Darstellungen führen. Welche Formatierungen die Browser unterstützen, finden Sie unter *https://caniuse.com*.

1.3 Dokumententyp und Zeichensatz

Der HTML5-DOCTYPE wird wie folgt angegeben (Groß- und Kleinschreibung spielt keine Rolle):

```
<!DOCTYPE html>
```

Obwohl in HTML5 fast alles erlaubt ist, was auch in HTML 4.01 und XHTML 1.0 erlaubt war, wurden doch einige Elemente und Attribute aus dem Standard entfernt. Wieder andere Elemente haben eine neue semantische Bedeutung erhalten. Eine Übersicht über die Unterschiede zu früheren Versionen finden Sie unter *https://www.w3.org/TR/html5-diff/*.

Unicode-Zeichensatz UTF-8

Als Zeichenkodierung für Ihre HTML- und CSS-Dateien sollten Sie UTF-8 verwenden, um möglichst flexibel zu sein.

Dazu notieren Sie Folgendes nach dem öffnenden Tag `<head>`:

```
<meta charset="utf-8">
```

Im folgenden ersten Beispiel sind Dokumententyp und Zeichenkodierung angegeben. Im weiteren Verlauf dieses Buches wird zugunsten der Übersichtlichkeit bei den **abgedruckten Quelltexten** auf die Angabe des DOCTYPE und des Zeichensatzes verzichtet. **Diese sollten Sie in realen Webseiten aber immer notieren!**

Die Beispieldateien zu diesem Buch sollen valide sein und funktionieren. Daher werden Sie in den eigentlichen Dateien alle notwendigen Angaben vorfinden.

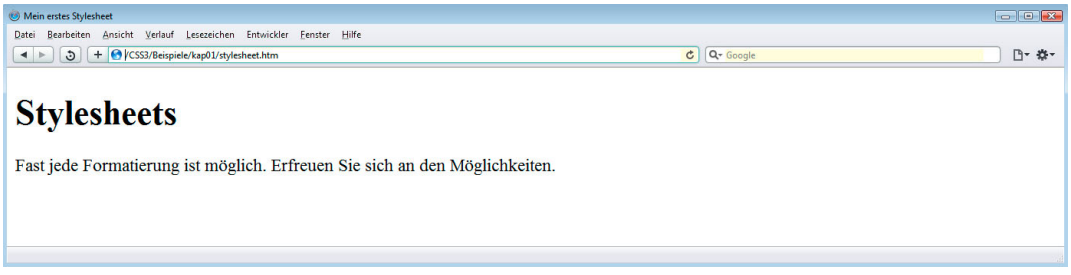
1.4 Formatvorlagen

Beispiel: *kap01\stylesheet.htm*

- Erstellen Sie in einem Texteditor das folgende HTML-Dokument.

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="utf-8">
  <title>Mein erstes Stylesheet</title>
</head>
<body>
  <h1>Stylesheets</h1>
  <p>Fast jede Formatierung ist möglich. Erfreuen Sie sich an den
     Möglichkeiten.</p>
</body>
</html>
```

- Speichern Sie den Quellcode als Datei *stylesheet.htm*.
- Öffnen Sie die Datei im Browser.



HTML-Dokument: Darstellung unter Verwendung der Browser-Formatvorlage

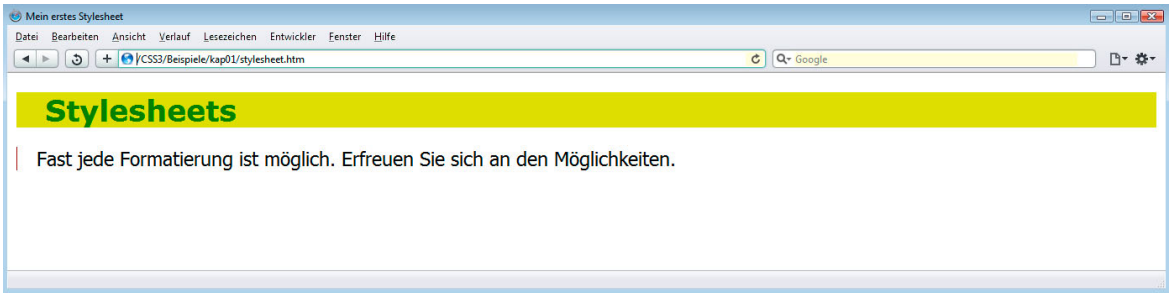
- Fügen Sie in die Datei nach dem Titel der Webseite die nachfolgend grau unterlegten Zeilen mit den Stilangaben ein.

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="utf-8">
  <title>Mein erstes Stylesheet</title>
  <style>
    h1 { font-family: Verdana, Arial, sans-serif;
         font-size: 1.5em;
         color: #00ff00;
         background-color: #dddd00;
         text-indent: 1em; }
    p  { font-family: Tahoma, Arial, sans-serif;
         border-left: 1px solid #880000;
         text-indent: 1em; }
  </style>
</head>
<body>
  <h1>Stylesheets</h1>
  <p>Fast jede Formatierung ist möglich. Erfreuen Sie sich an den
     Möglichkeiten.</p>
</body>
</html>
```

Die eingefügten Zeilen beinhalten Formatierungsanweisungen. Die verwendete Sprache, in der sie notiert werden, heißt Cascading Style Sheet (CSS). So entstehen Formatvorlagen, die denen eines Textverarbeitungsprogramms entsprechen. Das Beispiel enthält Regeln zur Darstellung von zwei HTML-Elementen: `p` (Absätze) und `h1` (Überschriften erster Ordnung). Wird eines dieser Elemente verwendet, wird es vom Browser den hier notierten Vorgaben entsprechend dargestellt.

Farben und Hintergrundfarben sind besonders deutliche Möglichkeiten der Hervorhebung. In den Beispielen ist es nicht möglich, ohne sie auszukommen. Erst durch Farben werden zum Beispiel die Dimensionen von Elementen sichtbar. In diesem Buch wird in der Regel die übliche hexadezimale Schreibweise für Farbwerte verwendet. Wie Farben notiert werden, wird in Kapitel 5 erklärt.

- Speichern Sie die Änderung und öffnen Sie die Datei in einem Browser.



HTML-Dokument: Darstellung unter Verwendung der Browser-Formatvorlage und einer Autoren-Formatvorlage

Das Beispiel beinhaltet Formatierungsregeln für zwei verschiedene Elemente, nämlich `h1` und `p`. Schauen Sie sich die Definition für `h1` etwas genauer an.

```
h1 { font-family: Verdana, Arial, sans-serif;
     font-size: 1.5em;
     color: #00ff00;
     background-color: #dddd00;
     text-indent: 1em; }
```

Diese Zeilen teilen dem Browser mit, dass eine Überschrift erster Ordnung folgendermaßen darzustellen ist:

- Schriftart (`font-family`): Verdana, Arial oder eine andere serifenlose Standardschrift (`sans-serif`)
- Schriftgröße (`font-size`): 1.5em (`em` = Breite des Buchstaben „m“. `1.5em` entspricht 150 % der normalen Schriftgröße)
- Textfarbe (`color`): grün
- Hintergrundfarbe (`background-color`): gelbliche Tönung
- Texteinzug (`text-indent`): `1em`

1.5 Stylesheets einbinden

Stylesheet-Angaben können auf drei verschiedene Arten in ein HTML-Dokument eingebunden werden:

- per Attribut in einem öffnenden Tag (`style="Eigenschaft:Wert;"`),
- im Kopf des HTML-Dokuments als Inhalt des Elementes `style`,
- in einer externen Datei. Der Pfad dieser Datei sollte dem Browser mittels `link`-Element mitgeteilt werden.

Stylesheets im Dokumentenkopf festlegen

Diese Methode kennen Sie bereits aus dem vorherigen Abschnitt. Beim Erstellen des ersten Dokuments haben Sie die Stylesheets im Kopf des Dokuments definiert.

```
<style></style>
```

Dies ist das Grundgerüst zum Definieren von Stylesheets. Der HTML-Tag `<style>` leitet die Formatdefinitionen ein.

Stylesheet-Angaben, die auf diese Weise im Kopf der HTML-Datei definiert werden, sind nur für diese eine HTML-Datei gültig. Damit verlieren Sie einen Hauptvorteil von Stylesheets, nämlich die Möglichkeit, das Aussehen vieler Seiten über eine einzige externe Datei zentral zu steuern (zentrale Formate). Möchten Sie mehreren Dateien dieselben Eigenschaften zuweisen, müssen Sie die Stylesheets in einer externen Datei definieren und in die HTML-Dateien einbinden.

Die Beispiele in diesem Buch zeigen, wann es sinnvoll ist, CSS im HTML-Head eines Dokumentes zu notieren: wenn Sie mit nur einer einzigen Datei Inhalte formatiert weitergeben oder präsentieren wollen. Da ein Webauftritt in aller Regel aus vielen Seiten besteht, sollten Sie die Formate in eigenständigen Dateien notieren. Der Browser wird diese Dateien laden und abspeichern. Alle Seiten können auf diese Datei zugreifen. So können alle Formate zentral gepflegt und verwendet werden. Sie müssen nur ein einziges Mal und nicht mit jeder Einzelseite neu übertragen werden.

CSS ist eine eigene Sprache. Innerhalb des Style-Elementes (also zwischen `<style>` und `</style>` dürfen keine HTML-Tags benutzt werden.

Stylesheets in eine externe Datei auslagern

Statt das Layout des Dokumentes immer wieder in allen HTML-Dokumenten festzulegen, können Sie eine zentrale Datei mit den Formatierungsanweisungen anlegen und von jeder einzelnen Webseite darauf zugreifen. Diese externe CSS-Datei enthält die Definitionen für alle Seiten Ihrer Website. Ändern Sie beispielsweise in dieser Datei die Angaben zur Schriftart, wird die Änderung in allen Dokumenten berücksichtigt, welche diese Vorlage verwenden.

Bei der CSS-Datei handelt es sich um eine **reine Textdatei**, die Sie in jedem Texteditor bearbeiten können. Verwenden Sie als Zeichensatz UTF-8. Die Endung des Dateinamens sollte `css` lauten (nicht zwingend erforderlich).

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

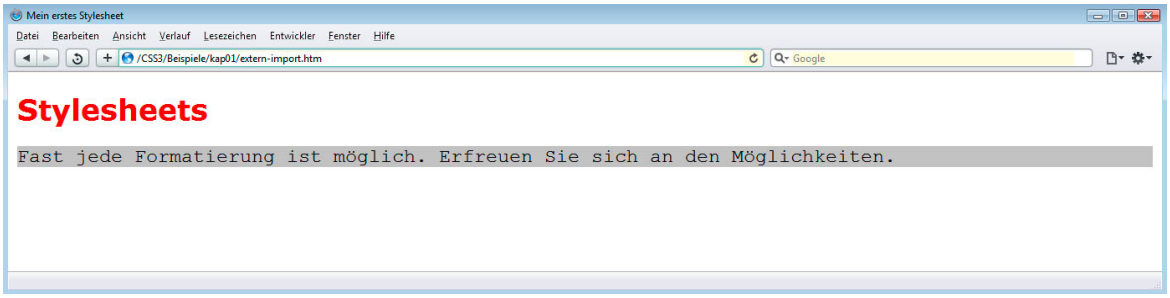
Diese Zeile fügen Sie in den Kopfbereich (`head`) der Webseite ein. Damit teilen Sie dem Browser mit, dass er die hier genannte Datei zur Darstellung des aktuellen HTML-Dokumentes verwenden soll. Der HTML-Tag `<link>` verweist auf eine externe Datei und `rel="stylesheet"` stellt den Bezug zu einem Stylesheet (Formatvorlage) dar. Das Attribut `href` enthält den Pfad zur gewünschten Datei.

- Erstellen Sie die nachfolgende CSS-Datei mit den Formatdefinitionen:

```
h1 { font-family: Verdana, Arial, sans-serif;
     font-size: 1.5em;
     color: red; }
p  { font-family: "Courier New", monospace;
     background: silver; }
```

- Speichern Sie die Datei unter dem Namen *layout.css*.
- Öffnen Sie in einem Texteditor das HTML-Dokument, das Sie bereits mit Stylesheets formatiert haben.
- Löschen Sie alle Zeilen, die Formatdefinitionen zwischen `<style>` und `</style>` enthalten.
- Fügen Sie folgende Zeile ein:
  `<link rel="stylesheet" href="layout.css">`

- Speichern Sie das HTML-Dokument *extern-import.htm* im selben Verzeichnis wie die Datei *layout.css*.
- Öffnen Sie das HTML-Dokument in Ihrem Browser.



HTML-Dokument mit externer Formatvorlage (kap01\extern-import.htm)

Die erstellte Formatdatei *layout.css* können Sie auch in andere HTML-Dokumente einbinden. Die Informationen innerhalb der Elemente `h1` und `p` erhalten dort ebenfalls dieselbe Formatierung.

Ändern Sie die Schriftgröße von Absätzen `p` auf `font-size: 75%`, ändert sich die Darstellung automatisch in allen Webseiten, in denen die CSS-Datei eingebunden ist. Ohne zentrale Vorlage müssten Sie jedes einzelne HTML-Dokument öffnen und die Schriftgröße für jeden Absatz oder wenigstens einmal pro Dokument anpassen. Mit einer zentralen Vorlage sind umfangreiche Änderungen am Gesamtlayout Ihrer Webseite effizient und konsistent umsetzbar.

Stylesheets importieren

Eine weitere Möglichkeit ist das Importieren von Stylesheets. Dies funktioniert grundsätzlich genauso wie das Laden aus einer Datei, jedoch über die CSS-spezifische Syntax. Dieser Import muss vor allen anderen CSS-Definitionen erfolgen, ansonsten werden die importierten Stylesheets nicht in das HTML-Dokument implementiert.

```
<style>
  @import url(dateiname.css);
</style>
```

Die Importdirektive zum Laden der Formatvorlage wird im HTML-Dokument im Style-Element notiert. Die Funktion `@import url()` lädt die angegebene CSS-Datei und stellt die Formatanweisungen für die Webseite zur Verfügung. Diese Methode ist weniger performant als das Einbinden per Link-Element. Es handelt sich hierbei um CSS-Syntax, daher muss die Angabe im Style-Element notiert werden. Als CSS-Angabe kann diese auch **am Anfang** einer CSS-Datei verwendet werden. So können Sie in eine Datei Formate importieren. Einige Entwickler trennen lange Formatvorlagen in mehrere Dateien auf, z. B.: `header.css`, `navigation.css`, `footer.css` usw.

! Für jede weitere Datei ist eine zusätzliche Anfrage an den Webserver nötig (http-Request). Jede Anfrage kostet Zeit!

Elemente mittels Attribut formatieren

Sie können jedem Element auch direkt im HTML-Code eine Formatierung zuweisen. Dazu geben Sie die Stylesheet-Deklarationen als Wert eines Attributes für das zu formatierende Element an.

```
<Element style="Eigenschaft:Wert;"> ... </Element>
```

Die Formatierung wird als Attribut `style` dem entsprechenden Element hinzugefügt. Zu beachten ist hierbei, dass Sie die Anführungszeichen setzen müssen.

Beispiel: *kap01\style-inline.htm*

```
<p style="font-family:Verdana,Helvetica,Arial,sans-serif;
color:#0000ff>Text</p>
```

Diese Methode hat gravierende Nachteile und sollte möglichst vermieden werden: Denn die konzeptionelle Trennung von Layout und logischer Strukturierung der Inhalte wird damit unterlaufen. Auf Englisch nennt man diese Art der Formatierung „presentational markup“.

Alle Elemente, die formatiert werden sollen, müssen das Style-Attribut mit entsprechenden CSS-Regeln erhalten und bei jeder Änderung müssen alle Elemente wieder angefasst werden. Weil Sie dabei leicht ein Element vergessen können, ist dieses Vorgehen fehleranfällig und erschwert die Wartbarkeit einer Website.

Die vielen unnötigen Wiederholungen aller Angaben müssen bei jedem Dateiabruf mit übertragen werden. Das kostet Performanz und Effizienz und damit Geld und Energie.

1.6 Rangfolge der Stylesheets

CSS bietet die Möglichkeit, mehrere Formatvorlagen zu nutzen. Unabhängig davon, ob Sie Formate im Kopfbereich der Webseite definieren, direkt am Element angeben oder von einer externen Datei einbinden: Die einzelnen Formatierungen können sich gegenseitig überschreiben. Diesem Prinzip der **Kaskadierung** verdankt CSS seinen Namen. Damit Sie das Überschreiben beeinflussen können, existiert eine Rangfolge.

Grundsätzlich gilt: Die zuletzt vom Browser bearbeitete Formatierung überschreibt frühere Formatangaben. Von der Reihenfolge unabhängig überschreiben genauere (spezifischere) Angaben ungenauere Angaben (vgl. Kapitel 2).

Im `head`-Bereich einer Webseite können Sie eine externe CSS-Datei einbinden und zusätzlich lokale Definitionen festlegen. Hier ist die Reihenfolge der Einbindung wichtig. Nutzen Sie z. B. eine extern bereitgestellte CSS-Datei und möchten Sie Änderungen nur für die Einzelseite vornehmen, müssen Sie zuerst die externe Datei einbinden und dann die lokale Anpassung vornehmen. Der Browser geht bei der Bearbeitung linear vor, das heißt, er beginnt am Anfang der HTML-Datei und arbeitet sich Zeichen für Zeichen weiter durch alle Angaben.

Wenn Sie zuerst die externe Datei einbinden und erst danach im Style-Element Angaben für die Einzelseite machen, überschreiben Sie damit **für diese eine Seite** die zuvor geladenen Angaben aus der externen Datei.

```
<link rel="stylesheet" href="dateiname.css">
<style>
/* Notieren Sie hier Ihre Formatierungsanweisungen */
</style>
```

Wenn Sie die externe Datei erst nach der lokalen Definition einbinden, können Sie lokale Änderungen vornehmen und werden womöglich keine Änderung sehen. Im ungünstigsten Fall würde Ihnen die externe Formatierung immer Ihre lokale Festlegung überschreiben.

Die Angabe von Formatierungen über das **Attribut** `style` überschreiben immer die Formatierungen der externen Formatierungen und die der Festlegungen im Kopfbereich der Webseite, weil diese Angaben sehr spezifisch sind (sie gelten genau für ein einziges Element).

Folgende Reihenfolge sollten Sie daher konsequent einhalten:

Externe Datei	`<link rel="stylesheet" href="dateiname.css">`
Lokale Definition	`<style></style>`
Definition am Element	`<Element style="">`

Beispiel: *kap01\extern.css*

In einer externen CSS-Datei legen Sie fest, dass die Überschrift `h3` rot darzustellen ist.

```
h3 { color: red; }
```

Beispiel: *kap01\rangfolge1.htm*

Es wird eine externe Datei eingebunden. Danach machen Sie Angaben, die nur für die aktuelle Seite gelten. Zuletzt überschreiben Sie diese im folgenden Beispiel mit Formatangaben direkt am Element.

```
<html lang="de">
<head>
  <title>Rangfolge von Stylesheets</title>
①  <link rel="stylesheet" href="extern.css">
②  <style>
     h3 { color: green; }
   </style>
</head>
<body>
③  <h3>Externe vor interner Definition: Farbe ist grün</h3>
④  <h3 style="color:blue;">Style überschreibt die vorherigen
Definitionen: Farbe wird blau</h3>
</body>
</html>
```

① Die externe CSS-Datei mit der Definition der roten Überschrift `h3` wird eingebunden.
② Im Kopfbereich legen Sie die lokale Definition für diese Webseite fest. In diesem Fall wäre `h3` in grüner Schriftfarbe darzustellen.
③ Die Ausgabe der Überschrift erfolgt in grüner Schrift.
④ In dieser Überschrift, die eigentlich grün sein soll, überschreiben Sie die Farbe mit dem Attribut `style`. Die Überschrift wird blau angezeigt.

Wenn Sie die externe Datei nach der lokalen Definition einbinden, also Punkt ① und ② vertauschen, wird die Überschrift ③ rot dargestellt. Die Überschrift ④ bleibt blau, da `style` für das Element `h3` zuletzt angegeben wird.

Spezifität einer Angabe erhöhen

Die Angabe von

```
!important
```

setzt die Spezifität einer Angabe sehr hoch. Sie legen damit fest, dass eine Formatierung nicht von nachfolgenden Angaben überschrieben werden soll. Dieses Schlüsselwort darf jedoch nicht für eine komplette Regel angegeben werden, sondern muss für jede CSS-Deklaration einzeln notiert werden, z. B.

```
h3 { color: green !important; font-size: 1em !important; }.
```

Setzen Sie `!important` bewusst und sparsam ein. Wenn Sie ein Format wie `color: green` notieren und der Browser das nicht wie gewünscht umsetzt, versuchen Sie, den Grund zu verstehen. Vielleicht haben Sie an späterer Stelle `color: black` notiert? Verwenden Sie `!important` nicht, weil es einfacher ist – Ihr Code wird schlechter verständlich und die Probleme mehren sich, wenn Sie nicht verstehen, warum der Browser nicht wie gewünscht reagiert.

CSS wurde entwickelt, um Layout und Inhalt voneinander zu trennen. Webseiten sind sehr viel einfacher zu gestalten und zu warten, wenn Sie von Anfang an konsequent alle Formate in einer zentralen Datei ablegen.

1.7 Ausgabemedien

Sie können Stylesheets für bestimmte Ausgabemedien definieren. In der Praxis werden beispielsweise oft unterschiedliche Angaben gemacht, je nachdem ob ein Dokument auf einem herkömmlichen Computer mit großem Bildschirm oder einem Smartphone angezeigt wird. Auch für den Druck werden meist Anpassungen vorgenommen.

Sinnvolle Angaben für Druckformate werden in einem gesonderten Kapitel am Ende des Buches besprochen.