Handout Webtypografie

Oliver Stickel

Sommersemester 2010

Das vorliegende Handout soll einen kurzen, grundsätzlichen Überblick über „Webtypografie“ sowie deren technische Grundlagen und Problemstellungen ermöglichen. Es ist im Kontext der zugehörigen Bildschirmpräsentation zu sehen, daher wird hier auf Bildmaterial verzichtet um das Dokument nicht unnötig aufzublähen.

1 Technik

1.1 Grundlagen

Wer sich mit Typografie beschäftigt, der muss zuerst die technischen Gegebenheiten unter denen die Schrift dargestellt wird, kennen. Im Falle von Webtypografie bezieht sich dies primär auf Bildschirme.

Grundsätzlich arbeitet ein solcher Bildschirm mittels eines (festgelegten) Rasters. Das Raster besteht aus Quadraten oder Rechtecken und seine „Zellen“ bestimmen die kleinste Einheit, die der betreffende Monitor darstellen *kann* Diese Einheiten nennen sich *Dots*, weshalb Auflösungsfähigkeit auch gern in *dpi* (Dots Per Inch) angegeben wird.

Das zweite wichtige Element für Bildschirme sind die sogenannten *Pixel*. Ein Pixel ist die kleinste Einheit, aus der der Bildschirm in der gewählten Auflösung seine Bilder aufbaut. Zur Verdeutlichung: Wenn ein Monitor eine physikalisch maximal mögliche Auflösung von 1024*768 Pixeln hat, so ist in dieser Auflösung 1 Pixel = 1 Dot. Betreibt man ebendiesen Monitor nun mit 800*600 Pixeln, so wird ein Pixel aus mehreren Dots zusammengesetzt, die zwar bei mittlerer Vergrößerung als zusammengehörige Einheit erscheinen, es in Wirklichkeit aber nicht sind, wie sich bei starker Vergrößerung zeigt.

Das heißt:

- Pixel und Dots sind keine absoluten Größen.
- Jedes Bild auf einem Monitor wird dargestellt durch „Kästchen“, die ab einer gewissen Distanz für unser Auge nicht mehr als solche erkennbar sind, sondern ein zusammenhängendes Bild ergeben.

Ein Buch hat ca. 300dpi, ein modernes Smartphone maximal 200dpi und ein Durchschnittsmonitor zwischen 72 und ca. 120dpi. Schon aus diesen nackten Zahlen lässt sich ablesen, dass im Bereich der Webtypografie die vergleichsweise sehr geringe Auflösungsfähigkeit von Bildschirmen unbedingt in Betracht gezogen werden muss.

1.2 Schriften

Aus der oben erwähnten Auflöseproblematik ergibt sich, dass der Schriftsatz auf dem Bildschirm diesem angepasst werden muss. Im klassischen Druck werden gerne Serifenschriftarten (wie beispielsweise Times New Roman) verwendet, da die Serifen den Lesefluss unterstützen und das Auge „führen“. Ein Buchdruck kann die feinen Linien der Serifen hierfür ausreichend gut darstellen, während dies auf Bildschirmen oft nicht möglich ist. Dies resultiert in einem „Verwaschen“ der Serifen, was das Lesen erschwert und natürlich unschön wirkt.

Primär werden im Bildschirmbereich daher sogenannte Sans-Serif oder Groteskschriften verwendet, die feine Linien vermeiden und auch bei geringer Auflösung klar lesbar sind. Auch solche Schriften lassen sich noch weiter für den Bildschirm optimieren, ein Beispiel hierfür ist die Verdana: Abstände zwischen Buchstaben und Wörtern sind sehr großzügig um auch bei sehr kleinen Schriftgrößen Lesbarkeit zu garantieren, die Buchstaben I J l und die Zahl 1 sind klar unterscheidbar, die Pixelmuster der Schrift wurden üblichen Bildschirmcharakteristika angepasst und insgesamt ist die Schrift sehr auf Skalierbarkeit ausgelegt. Dies ist nötig, da wir auf dem Bildschirm enorm unterschiedliche Schriftgrößen haben, die es im Buchdruck so flexibel nicht gibt.

All dies bedeutet nicht, dass Serifen im Web keinen Platz hätten. Sie werden beispielsweise gerne als Stilmittel verwendet oder aber bei Inhalten, von denen man davon ausgeht, dass sie ausgedruckt werden (wissenschaftliche Arbeiten im pdf-Format oder vergleichbares).

2 Software-Ebene

2.1 HTML - HyperText Markup Language

Da es hier um *Web*typografie geht, ist ein kleiner Abriß über HTML unabdingbar: Diese Sprache ist immerhin Grundlage des Web wie wir es heute als User in erster Linie kennen. HTML wurde 1989 erfunden, ist zum Standard im Web geworden und wird seitdem konstant weiterentwickelt und erweitert. Die wichtigste Idee hinter HTML sind die Hyperlinks. Dass diese „Links“ uns heute als ganz normal oder gar trivial erscheinen, ist letztlich ein Beleg dafür, wie außerordentlich erfolgreich dieses Konzept ist, das effektiv einen der Grundpfeiler des Web bildet.

HTML ist eine sogenannte Markup-Sprache. Das heißt, es gilt nicht das WYSIWYG-Prinzip („What you see is what you get“), wie es beispielsweise in Officeprogrammen wie

Microsoft Word der Fall ist. In HTML werden durch sogenannte *Markups* oder *Tags* bestimmte Textbereiche voneinander abgegrenzt und bekommen Eigenschaften zugewiesen. So ist ein Textbereich, der in der Umgebung <h1>TEXT</h1>steht, als Überschrift definiert und wird entsprechend dargestellt. Um noch einmal das Gegenbeispiel zu bemühen: In MS Word würde man den gewünschten Text markieren, entsprechend formatieren und sähe sofort das Resultat: WYSIWYG.

Wichtig ist hierbei: Wie *im Detail* z.B. besagte Überschrift dargestellt wird, ist nicht festgelegt, sondern wird erst im Webbrowser des Users definiert. Ob die Überschrift nun in Schriftgröße 20, Schriftart Times New Roman und fett oder aber in Schriftgröße 16, Schriftart Tahoma und kursiv dargestellt wird, ist reine Einstellungssache und hat mit dem HTML Code nichts zu tun. Dieser definiert nur die Textauszeichnung, die man flapsig als „Dies ist eine Überschrift“ bezeichnen könnte.

2.2 Unicode

Unicode stellt die Bemühung dar, alle Textelemente (weltweit) in digitale Repräsentationen zu überführen. Dazu gehören offensichtlich Buchstaben und Zahlen, aber auch Leerzeichen unterschiedlicher Größe, Symbole aller Art, (Nicht-)Trennzeichen, etc.
Beispiele hierfür sind unterschiedliche Anführungszeichen: „und“, nicht zu verwechseln mit ”,welches direkt über die Tastatur (über die Tastten Shift+2) erreichbar ist. Moderne Textverarbeitungsprogramme ersetzen dies mitunter automatisch, in der Webtypografie kann man sich hierauf nicht jedoch nicht verlassen.
Weitere Beispiele sind Striche: Halbgeviert „12–17 Uhr“, Viergelgeviert: „H-Milch“ und das mathematische Minus „−“. All diese Zeichen haben ihre Unicode-Repräsentationen, z.B. das Minus als U+2212 (hexadezimal) und sollten entsprechend eindeutig verwendet werden.
In die Kategorie der nicht direkt sichtbaren Textelemente fällt der „Soft Hyphen“ U+00AD. Dieser setzt in Texte mögliche Umbruchstellen ein, damit längere Wörter beim (Nicht-)Umbrechen das Layout nicht „zerstückeln“.

2.3 CSS - Cascading StyleSheets

An der o.g. Interpretation von Markups durch den Browser setzt CSS an: HTML selbst bietet hier, auch historisch bedingt, nur eingeschränkte Layoutmöglichkeiten, die dem anspruchsvolleren Layouter und dem Zeitgeist leider nicht mehr genügen. Daher „manipuliert“ CSS die im Browser dargestellten Inhalte und erweitert die Darstellungsmöglichkeiten ausgesprochen vielfältig.

CSS benutzt ein Box-Modell um Inhalte wie Texte, Bilder, etc. flexibel absolut oder relativ zu positionieren und auch von der Darstellung her zu beeinflussen. Auch im Sinne einer „schönen“, wartungsfreien und flexiblen Programmierung ist diese Trennung von Layout und Inhalt anzustreben, da so der zugrundeliegende Code nicht wirklich verändert wird: Aus einem HTML-Dokument können ganz unterschiedlich aussehende

Seiten erstellt werden, oder aber es werden verschiedene HTML-Seiten automatisiert und einfach durch eine CSS-Datei mit dem gleichen Layout versehen.

2.3.1 Dynamik

Zu beachten ist die hohe Dynamik und die ganz unterschiedlichen Endgeräte auf denen Web-Inhalte konsumiert werden: Ein Text auf einem Smartphone wird ganz unterschiedlich dargestellt und umgebrochen als auf einem 24-Zoll-Monitor. Entsprechend flexibel und durchdacht müssen Inhalte konzipiert, entwickelt und implementiert werden. Auch dabei kann CSS zum Teil helfen:

Exemplarisch seien hier Zeilenhöhen genannt: Es macht Sinn, einen Zeilenrhythmus einzuhalten. Der Text wirkt so ruhiger, besser strukturiert und letztlich besser lesbar. Um das zu konkretisieren: Wenn wir z.B. davon ausgehen, dass eine Standardüberschrift 18px hat, ein Text 12px, setzen wir uns eine Zeilenhöhe von 18px, an der sich der gesamte Text orientiert und erzielen so einen gut lesbaren Text. Es gibt natürlich noch weitere Optimierungsmöglichkeiten (Einzüge an Abschnittsanfängen, etc.), auf alle einzugehen würde hier jedoch den Platz sprengen.
Die 18px-Zeilenhöhe lässt sich auch dynamisch als $18/12 = 1,5em$ mit $1em = 12px$ darstellen, was die Darstellung auf verschiedenen Endgeräten vereinheitlicht. Die Einheit *em* kann man sich in diesem Kontext als Standardschriftgröße, die im Browser des Users eingestellt ist, vorstellen. Wenn sie also dank einer solchen lokalen Einstellung von 12px abweicht, passt sich die Zeilenhöhe verhältnismäßig betrachtet an. Diese „Dynamisierung“ muss natürlich nicht nur auf die Zeilenhöhe beschränkt bleiben, sondern lässt sich noch weiter treiben um möglichst größe Dynamik zu erzielen.

3 Ausblick

3.1 Spalten

Als weiteres Beispiel für die Lesbarkeit und die Dynamik sei hier die Unterteilung in Spalten genannt, wie man sie aus der Zeitung kennt: Dies macht nicht nur ästhetisch Sinn, sondern fördert auch den Lesefluss. Durchschnittlich beträgt die Breite der Zone in der seitens des menschlichen Auges am Bildschirm bequem fokussiert werden kann lediglich 8cm (Faustregel, nicht exakt). Das ergibt eine durchschnittliche Spaltenbreite von circa 60-70 Zeichen für normalen Fließtext für optimales Lesen. Spaltensatz ist im Web bisher aber nur durch Tricks möglich, weshalb die kommende CSS-Version 3 eine direkte, echte Möglichkeit hierfür bieten wird.

3.2 Webfonts

Aus der Darstellung die erst am eigenen PC entsteht ergibt sich auch, dass jede Schrift, die vom Layouter verwendet wird auch auf dem Rechner des Users vorhanden sein muss, sonst wird sie nicht korrekt dargestellt. Historisch bedingt sind leider nur eine (kleine) Hand voll Schriften so verbreitet, dass man sie als Layouter gefahrlos einsetzen kann.

Das ist weder ästhetisch besonders befriedigend noch typografisch wirklich sinnvoll.

Als Umgehung des Problems wurde bisher oft getrickst und anstatt „echter“ Schrift einfach ein Bild mit dem gewünschten Schriftmotiv eingebunden. Diese Praktik ist extrem unschön, da die Bilder je nach Auflösung, Vergrößerung oder Endgerät extrem unterschiedlich und teilweise sehr unschön verzerrt oder größenverändert werden können. Davon ganz abgesehen ist die Barrierefreiheit hier nicht gegeben. So kann eine Vorlesesoftware für Blinde aus einem Bild keinen Text herauslesen und teilt dem Zuhörer statt der möglicherweise wichtigen abgebildeten Überschrift lediglich mit, dass hier ein Bild zu sehen ist.

In neuerer Zeit sind zur Umgehung dieses Problems die Techniken sIFR und Cufón aufgekommen. Beide basieren auf dem Prinzip, die gewünschte Schrift im HTML-Code auszulesen und durch Flash-Elemente oder Vektorgrafiken zu ersetzen, welche dynamisch erzeugt werden, also skalierbar sind. Ersteres macht Probleme wenn auf dem Browser kein Plugin zur Darstellung von Flash installiert ist und beide Techniken haben unter Umständen Probleme mit bestimmten Browsern oder zumindest Browserversionen. Davon abgesehen lassen sich schöne, dynamische Resultate erzielen und die Barrierefreiheit ist gegeben, da der Original-HTML-Code noch existiert und ausgelesen werden kann. Er wird ja nur lokal im Browser der jeweiligen User in der Darstellung ersetzt.

CSS Version 3 wird alle Umgehungstaktiken unnötig machen, da es die sogenannten Webfonts beinhalten wird. Hier werden die Schriftarten direkt via CSS in die zu erstellende Website eingebunden. Man kann sich das Verfahren mehr oder minder vorstellen wie die Einbindung verwendeter Schriftarten in eine Powerpoint-Präsentation die auf einem fremden Rechner gehalten werden soll. Dies stellt die korrekte Darstellung an jedem Ort sicher. Code-technisch reduziert sich das ganze Prozedere auf ca. zwei Zeilen, ist also deutlich einfacher als die bisherigen Verfahren. Auch hier muss natürlich der Browser den neuen CSS3-Standard unterstützen, was bei älteren Browsern unter Umständen Probleme bereiten könnte.