

Inhalt

1	Denken in flexiblen Strukturen	21
1.1	Responsive Webdesign: Was bedeutet das eigentlich?	21
1.1.1	Veränderte Anforderungen an die Darstellung von Websites	22
1.1.2	Anpassungsfähige Websites versus Mobilversionen von Websites	25
1.1.3	Beispiele für anpassungsfähige Websites	26
1.2	Layouttypen, feste, fluide und flexible	29
1.2.1	Das feste Layout	29
1.2.2	Das fluide und das elastische Layout	29
1.2.3	Das adaptive Layout	30
1.2.4	Das responsive Layout	30
1.3	Flexible Raster – Gridsysteme	31
1.3.1	Anwendungsbeispiel: Fixes Raster in flexibles umrechnen	32
1.4	Layoutumbrüche – Breakpoints	36
1.4.1	Adaptives oder responsives Layout	37
1.5	Zusammenfassung	38
2	Schnelleinstieg: Responsive Umsetzung eines fixen Layouts	39
2.1	Die Ausgangslage: Ein grafischer Entwurf mit festen Abmessungen	39
2.2	Der erste Schritt: Feste Raster in flexible umrechnen	43
2.3	Der zweite Schritt zu mehr Flexibilität: Anpassungsfähige Inhalte	46
2.3.1	Exkurs: Flexible Bilder	47
2.4	Der dritte Schritt: Layouts mit Media Queries umschalten	49
2.4.1	Exkurs: Media Queries	49
2.5	Zusammenfassung	51
3	Die Schlüsseltechnologie Media Queries	53
3.1	Cascading Stylesheets (ein kurzer Rückblick)	54
3.1.1	Zuweisung von CSS-Eigenschaften	54

- 3.2 Medientyp (Media Type)** 55
 - 3.2.1 Medienabfrage für den Medientyp setzen 55
 - 3.2.2 Medientypen in der Übersicht 56
- 3.3 Medieneigenschaften (Media Features)** 57
 - 3.3.1 Medieneigenschaften in der Übersicht 58
- 3.4 Media Queries schreiben** 60
 - 3.4.1 Komponenten eines Media Querys 60
 - 3.4.2 Media Queries zuweisen 62
- 3.5 Viewports und Pixel** 63
 - 3.5.1 Der visuelle Viewport 64
 - 3.5.2 Der Layout-Viewport auf mobilen Geräten 64
 - 3.5.3 Gerätepixel und CSS-Pixel: Der »virtuelle« visuelle Viewport 65
 - 3.5.4 Das Viewport-Metatag und seine Eigenschaften 68
 - 3.5.5 Die @viewport-Anweisung in CSS 70
- 3.6 Media Queries in der Praxis** 72
 - 3.6.1 Medieneigenschaft »width« 72
 - 3.6.2 Media Queries in em 73
 - 3.6.3 Medieneigenschaft »height« – vertikale Media Queries 74
 - 3.6.4 Medieneigenschaft »orientation« 75
 - 3.6.5 Medieneigenschaft »aspect-ratio« 76
 - 3.6.6 Medieneigenschaften »resolution« und »device-pixel-ratio« 76
 - 3.6.7 Medieneigenschaften »pointer« und »hover« 78
 - 3.6.8 Browserunterstützung und Fallbacklösungen 79
- 3.7 Media-Query-Unterstützung mit JavaScript** 80
 - 3.7.1 Element Queries und Container Queries 82
 - 3.7.2 Restive JS (rScript) für Geräteerkennungen 83
- 3.8 Serverseitige Geräte- und Feature-Erkennung** 83
 - 3.8.1 WURFL & Co. 84
 - 3.8.2 RESS – das Beste von Client und Server kombiniert 84
 - 3.8.3 Client Hints 85
- 3.9 Zusammenfassung** 86

- 4 Ein responsiver Workflow** 87
 - 4.1 Der alte Prozess** 87
 - 4.2 Phase 1: Moodboards und Inhaltsplan** 91
 - 4.2.1 Moodboards 91

4.2.2	Inhalte strukturieren und hierarchisieren	92
4.2.3	Content-Prototypen	93
4.3	Phase 2: Style Tiles und Wireframes	94
4.3.1	Style Tiles	94
4.3.2	Mockups	97
4.4	Phase 3: Design im Browser	101
4.4.1	Frameworks und Website-Editoren als Design-Tools	102
4.4.2	Komponenten, Patterns und Atomic Design	102
4.5	Phase 4: Rinse and Repeat	103
4.5.1	Beziehen Sie Ihre Kunden in den Gesamtprozess mit ein	103
4.6	Das responsive Team	105
4.7	Dokumentation responsiver Designs	106
4.7.1	Dokumentation mit Living Styleguides	107
4.8	Zusammenfassung	109
5	Design und Typografie	111
5.1	Design follows Content	111
5.1.1	You cannot not communicate – kein Design ist auch ein Design	112
5.2	Design von innen nach außen – der Atomic-Design-Ansatz	114
5.2.1	Atomic Design anwenden mit Pattern Lab	115
5.3	Designanforderungen für responsive Sites	121
5.3.1	Does size matter – was machen Nutzer mit ihren Geräten?	122
5.3.2	Geräteübergreifendes Surfen	124
5.3.3	Size matters: Ziele für Touchevents	125
5.3.4	Es gibt kein Hover auf Hawaii, und ein Klick ist kein Touch	126
5.3.5	Inaktives »:active« auf iOS	128
5.3.6	Handpositionen	129
5.3.7	Layoutwechsel bei Änderung der Orientierung	130
5.3.8	Schreiben ist mühsam: Formulare auf Smartphones	132
5.3.9	Mobile Inspiration und Best Practice	133
5.4	Typografie (anpassungsfähiger Text)	134
5.4.1	Die Auswahl der Schriftart	134
5.4.2	Angaben für die Schriftgröße	137
5.4.3	Schriftgrößenwahrnehmung auf kleinen und großen Bildschirmen	142
5.4.4	Zeilenlänge und Durchschuss	144

5.4.5	Praxisbeispiel: Relative Schriftgrößen bezogen auf die Viewport-Breite	146
5.4.6	Praxisbeispiel: Mehrspaltensatz	148
5.4.7	Silbentrennung und Textbeschnitt	149
5.5	Zusammenfassung	151

6 Semantik und Barrierefreiheit 153

6.1	Prinzipien der Zugänglichkeit	153
6.1.1	Wahrnehmbarkeit	154
6.1.2	Bedienbarkeit	159
6.1.3	Verständlichkeit	159
6.1.4	Robustheit	161
6.2	Semantik durch HTML5	161
6.2.1	Neue HTML-Elemente	162
6.2.2	HTML5-Formularattribute für mehr Semantik	165
6.3	Semantik durch WAI-ARIA-Rollen	167
6.4	Zusammenfassung	170

7 Responsive Layout-Patterns 171

7.1	Mobile First	172
7.1.1	Reduktion auf das Wesentliche ist die Devise	172
7.1.2	Mobile First – Progressive Enhancement für das Layout	173
7.1.3	Mobile First bedeutet auch Content First	174
7.2	Praxisbeispiel: Mobile First	174
7.2.1	Mobile First – los geht's!	174
7.2.2	Mockups für das Layout der Beispiel-Website	175
7.2.3	Basisversion: Smartphone-Ansicht	176
7.2.4	Setzen des Viewport-Metatags	184
7.3	Auswahl der Breakpoints	185
7.3.1	Haupt-Breakpoints	186
7.3.2	Anpassungs-Breakpoints	187
7.3.3	Vertikale Breakpoints	188
7.4	Praxisbeispiel: Ersten Breakpoint setzen (Tablets)	189

7.5	Layout-Patterns (Darstellungsmuster) für unterschiedliche Ausgabegeräte	193
7.5.1	Tiny Tweaks (kleine Optimierungen)	193
7.5.2	Mostly Fluid (größtenteils fließend)	194
7.5.3	Column Drop (abgesenkte Spalten)	195
7.5.4	Layout Shifter (Layoutverdreher)	196
7.5.5	Off-Canvas-Layout (außerhalb des Bildschirms)	197
7.5.6	Footer-Navigation und Off-Canvas-Marginalie	197
7.5.7	Top-Off-Canvas-Menü und Off-Canvas-Marginalie	198
7.5.8	Vertikale und horizontale Off-Canvas-Panels	199
7.5.9	Zusammengefasste Off-Canvas-Elemente	200
7.5.10	Off-Canvas-Lösungen aus der Schublade	201
7.6	Praxisbeispiel: Weitere Breakpoints setzen (große Screens)	202
7.6.1	Kleine Desktopversion nach dem Konzept Layout Shifter	202
7.6.2	Große Desktopversion	204
7.7	Flexbox-Layout	206
7.7.1	Umsetzung des Praxisbeispiels als Flexbox-Layout	206
7.7.2	Praxisbeispiel: Flexbox-Layout Content Switch	208
7.8	Grid-Layouts (CSS3)	211
7.8.1	Umsetzung des Praxisbeispiels als CSS-Grid-Layout	211
7.9	Zusammenfassung	218
8	Frameworks für responsives Design	219
8.1	Eigene Vorlage oder fertige Frameworks verwenden?	220
8.1.1	Gridpak zum Erstellen von Rastern mit Media Queries	220
8.1.2	HTML5-Boilerplate und Initializr	221
8.2	Wie wählen Sie das richtige Framework aus?	222
8.3	Eine kurze Vorstellung responsiver Frameworks	223
8.3.1	Foundation	224
8.3.2	Bootstrap	228
8.3.3	PureCSS	232
8.3.4	Simple Grid	235
8.4	JavaScript-Bibliotheken	237
8.4.1	jQuery	237
8.4.2	Modernizr	238

8.5	Elegante Stylesheets mit Präprozessoren: SASS & Co.	240
8.6	Zusammenfassung	247
9	Navigationskonzepte	249
9.1	Was macht eine Navigation benutzerfreundlich?	249
9.2	Benutzerfreundliche Navigation für mobile Geräte	250
9.2.1	Freier Blick auf die Website	251
9.2.2	Ausreichend große Klickflächen für die Touchbedienung	251
9.2.3	Umgang mit Menüs mit mehreren Ebenen	251
9.2.4	Navigationstypen für mobile Geräte mit Touchscreen	252
9.3	Wenige Menüpunkte am oberen Rand	252
9.3.1	Praxisbeispiel: Mininavigation – wenige Menüpunkte am oberen Rand	253
9.4	Lange Menüs kompakt anordnen	256
9.4.1	Praxisbeispiel: Priority-Menü	257
9.5	Select-Menü	260
9.5.1	Praxisbeispiel: TinyNav – Select-Menü	261
9.6	Navigation per Anker am Ende des Seiteninhalts	265
9.6.1	Praxisbeispiel: Footer-Navigation mit Anker	266
9.7	Toggle-Menü	269
9.7.1	Praxisbeispiel: Toggle-Menü mit dem Plug-in Responsive-Nav	269
9.8	Off-Canvas-Menü	272
9.8.1	Praxisbeispiel: Off-Canvas-Menü mit Pushy	273
9.8.2	Praxisbeispiel: Swipe-Menü mit Slideout.js	275
9.9	Multilevel-Menüs	280
9.9.1	Die native Einbindung von Multilevel-Menüs auf iOS und Android	280
9.9.2	Praxisbeispiel: Multilevel-Menü mit DoubleTabToGo.js	281
9.9.3	Praxisbeispiel: Multilevel-Menü mit Flexnav	282
9.9.4	Noch mehr Multilevel-Menüs ...	285
9.10	Zusammenfassung	285

10	Flexible Bildelemente	287
10.1	Anpassungsfähige Bilder	287
10.1.1	Praxisbeispiel: Anpassungsfähiges Headerbild	289
10.1.2	Bilder ausschnittsweise anzeigen	290
10.1.3	Praxisbeispiel: Bildausschnitt auf schmalen Screens	291
10.1.4	Flexible Bilder, die nicht über die ganze Rasterbreite gehen	293
10.1.5	Praxisbeispiel: Flexible Teaserboxen mit Bild und Text	293
10.2	Responsive Hintergrundbilder	296
10.2.1	Praxisbeispiel: Gekachelte Bitmap-Hintergrundmuster	297
10.2.2	Praxisbeispiel: Hintergrundmuster mit CSS3 erstellen	298
10.2.3	Praxisbeispiel: Hintergrundmuster mit SVG	300
10.2.4	Großflächige Hintergrundbilder	302
10.2.5	Praxisbeispiel: Website mit vollflächigen Hintergrundbildern	305
10.2.6	Praxisbeispiel: Vollflächige Hintergrundbilder in Teaserboxen	309
10.3	Responsive Icons	312
10.3.1	Vorbereitung für alle Praxisbeispiele zu responsiven Icons: Erstellung der Icon-Sets	313
10.3.2	Praxisbeispiel: Icons als einzelne CSS-Hintergrundbilder	315
10.3.3	Praxisbeispiel: Icons als CSS-Hintergrundbilder aus einer Sprite-Datei	319
10.3.4	Icon-Fonts	322
10.3.5	Praxisbeispiel: Responsive Icons als Icon-Font	326
10.3.6	Praxisbeispiel: Icon-Font mit Ligaturen	330
10.3.7	SVG-Icons (sind die beste Wahl)	332
10.3.8	Praxisbeispiel: Responsive Icons aus Inline-SVG-Sprite	332
10.3.9	Praxisbeispiel: Icons aus extern eingebundener SVG-Datei	337
10.3.10	Praxisbeispiel: Icons aus externer SVG-Datei über CSS einfärben	339
10.4	Auflösungsunabhängige Grafiken (SVG)	341
10.4.1	Das Scalable-Vector-Graphics-Format	341
10.4.2	Praxisbeispiel: SVG-Infografik versus GIF-Infografik	341
10.4.3	Einbindung von SVG-Grafiken	342
10.4.4	Praxisbeispiel: Unterschiedliche SVG-Einbindungsarten	348
10.4.5	Die responsive SVG-Einbindung	352
10.4.6	Praxisbeispiel: SVG-Filter	353
10.5	Die Syntax für responsive Bilder	355
10.5.1	Syntax für responsive Hintergrundbilder	356
10.5.2	Praxisbeispiel: Responsive Bilder mit »srcset« und »sizes«	357
10.5.3	Praxisbeispiel: Responsive Art-Direction – das <picture>-Element	361

10.5.4	Browserunterstützung und Fallbacks für responsive Bilder	364
10.5.5	Fazit zur responsiven Syntax	365
10.5.6	Mit Client Hints zu echten responsiven Bildern	365
10.6	Unterschiedliche Versionen für responsive Bilder erzeugen	367
10.6.1	Die Nulllösung: Hochauflösende und komprimierte Bilder	368
10.6.2	Automatisch Bilder und Code generieren mit dem Responsive Image Breakpoints Generator	369
10.6.3	Bilder mit Automatisierungstools generieren (Grunt, Gulp)	372
10.6.4	Bildvarianten auf dem Server generieren	372
10.6.5	Bilder von der Cloud ausliefern lassen	373
10.7	Zusammenfassung	377
11	Mehr flexible Inhalte	379
11.1	Responsive Slider nicht nur für Bilder	380
11.1.1	Praxisbeispiel: Bilder-Slider mit dem Slick-Plug-in	380
11.1.2	Praxisbeispiel: Bilderkarussell mit dem Slick-Plug-in	382
11.1.3	Praxisbeispiel: Textblock-Slider mit dem Slick-Plug-in	386
11.1.4	Weitere Bildergalerie-Tools	388
11.2	Responsive Lightboxen	389
11.2.1	Praxisbeispiel: Responsive Lightbox mit Fancybox	390
11.3	Responsive Image Maps	392
11.3.1	Praxisbeispiel: Flexible Image Map mit jQuery-rwdImageMaps.js	392
11.4	Anpassungsfähige Videos	394
11.4.1	HTML5-Video Unterstützung und Formate	394
11.4.2	Praxisbeispiel: HTML5-Videos	395
11.4.3	Praxisbeispiel: YouTube & Co. – Videos im iframe	397
11.4.4	Praxisbeispiel: Videoseitenverhältnisse mit FitVids.js und FluidVids.js	400
11.4.5	Praxisbeispiel: HTML5-Videoplayer mit video.js	401
11.5	Flexible Karteneinbindungen (Maps)	403
11.5.1	Praxisbeispiel: Google Map im iframe	403
11.5.2	Praxisbeispiel: Flexible Google-Map-Einbindung mit Google API	404
11.6	Flexible Tabellen	406
11.6.1	Praxisbeispiel: Datentabellen durch Scrollen zugänglich machen	407
11.6.2	Praxisbeispiele: Tabellen mit CSS(-generiertem Content) umstrukturieren	408

11.6.3	Praxisbeispiel: Anpassungsfähige Tabellen mit JavaScript-Plug-ins	410
11.7	Akkordeons und Inhaltsboxen mit Reitern	414
11.7.1	Praxisbeispiel: Tab-Reiter zu Akkordeon mit smartTabs.js	415
11.8	Flexible Formulare	418
11.8.1	Praxisbeispiel: Responsives Formular	418
11.9	Inhalte selektiv anzeigen und laden	423
11.9.1	Inhalte entfernen oder ergänzen – wann und wie?	424
11.9.2	Inhalte per CSS ausblenden (display: none)	424
11.9.3	Praxisbeispiel: Inhalte per CSS-generiertem Content hinzufügen	425
11.9.4	Praxisbeispiel: Inhaltsblöcke mit AppendAround neu anordnen	427
11.10	Flexible Werbung	430
11.10.1	Bewusstsein schaffen für die veränderten Rahmenbedingungen	430
11.10.2	Alte und neue Bannerkonzepte	431
11.10.3	Bannererstellung und -auslieferung	433
11.10.4	Fixe Spalte und nur ein Rectangle-Format	434
11.10.5	Gezieltes Laden von Bannergrößenformaten mit Lazy-Ads	435
11.10.6	ZURB-Playground: Responsive Ads	437
11.10.7	Google Responsive Ads	437
11.11	Responsive HTML-E-Mails	438
11.11.1	HTML-E-Mail-Templates responsiv einsetzen	439
11.12	Zusammenfassung	442
12	Testing und Qualitätssicherung	443
12.1	Validatoren für HTML und CSS	443
12.2	Breakpoints im Browser testen	444
12.2.1	Firefox	445
12.2.2	Chrome	445
12.2.3	Safari	446
12.2.4	Breakpoint-Tester	446
12.2.5	Pattern Lab als Testplattform	448
12.2.6	Der Mobile Emulator für Opera	448
12.2.7	Testen und Präsentieren	448
12.3	Auf mobilen Geräten testen	450
12.3.1	Open Device Labs weltweit	450
12.3.2	Das Home-Device-Lab	451

12.3.3	Testen in der Cloud	455
12.3.4	Real-Life-Testen mit »unkooperativen« Inhalten	459
12.4	Qualitätssicherung und Wartung	461
12.4.1	Dokumentationen erstellen und pflegen	461
12.4.2	Modularisierung und Benennungsschemata	462
12.4.3	Stylesheets knapp schreiben und schlank halten	464
12.4.4	Unbenutzte CSS-Deklaration und Klassen finden	467
12.4.5	Tools für die Automatisierung von Prozessen	468
12.5	Zusammenfassung	469
13	Performanceoptimierung	471
13.1	Das Performancebudget	472
13.2	Was beeinträchtigt die Performance?	473
13.2.1	Anatomie einer Website	474
13.3	Skripte, Stylesheets und HTML	476
13.3.1	Aus den Augen, aus dem Sinn?	477
13.3.2	Skripte zusammenfassen	479
13.3.3	Drittanbieterskripte und Social-Media-Buttons	481
13.3.4	CSS-Sprites und Data-URLs sparen Requests	485
13.3.5	Stylesheets und in ihnen verlinkte Ressourcen werden unterschiedlich geladen	486
13.3.6	Minifying und Dateikompression	487
13.3.7	Die Zukunft: HTTP/2 versus HTTP/1.1	488
13.4	Caching	489
13.5	Performanceoptimierung für Grafiken und Bilder	490
13.5.1	Optimierung von Bitmap-Bildern	490
13.5.2	SVG-Optimierung	491
13.6	Web-Schriften optimieren	492
13.6.1	Buchstabenauswahl verkleinern	492
13.6.2	Fonts direkt einbetten	494
13.7	Gefühlte Performance und Nachladen von Inhalten	495
13.7.1	Stylesheets an den Anfang, JavaScript-Dateien an das Ende der Webseite	495
13.7.2	Missionskritisches CSS (alias »above the fold«)	496
13.7.3	Praxisbeispiel: Lazy Loading von Bildern mit Lazy Sizes	498

13.7.4	Praxisbeispiel: Lazy Sizes zum Nachladen von Inhalten in Tab-Content	499
13.7.5	Praxisbeispiel: Conditional Loading Content via JavaScript (und CSS)	500
13.7.6	Farbflächen-»Vorschau« für Bilder	505
13.8	Zusammenfassung	507

14	Fazit	509
-----------	--------------------	------------

Anhang	511
Index	513