

深入浅出移动互联网 (影印版)

Head First Mobile Web



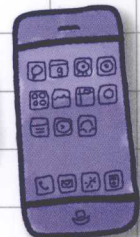
Build
once, run
everywhere



Be more supportive
(of your users)



Find your
way with
geolocation



Shape-shift your
sites with Responsive
Web Design

Put your
pages on a
small-screen diet



O'REILLY® 东南大學 出版社

Lyza Danger Gardner
& Jason Grigsby 著

深入浅出移动互联网 (影印版)

Head First Mobile Web

Wouldn't it be dreamy if there were a book to help me learn how to build mobile websites that was more fun than going to the dentist? It's probably nothing but a fantasy...



Lyza Danger Gardner
Jason Grigsby 著

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'Reilly Media, Inc. 授权东南大学出版社出版

东南大学出版社

图书在版编目 (CIP) 数据

深入浅出移动互联网: 英文 / (美)加德纳(Gardner, L.D.),
(美)克莱格斯(Grigsby, J.)著. —影印本. —南京: 东南
大学出版社, 2012.6

书名原文: Head First Mobile Web

ISBN 978-7-5641-3414-3

I. ①深… II. ①加… ②克… III. ①移动通信—互
联网—英文 IV. ①TN929.5

中国版本图书馆 CIP 数据核字 (2012) 第 065811 号

江苏省版权局著作权合同登记

图字: 10-2012-3 号

©2011 by O'Reilly Media, Inc.

Reprint of the English Edition, jointly published by O'Reilly Media, Inc. and Southeast University Press, 2012. Authorized reprint of the original English edition, 2011 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly Media, Inc. 出版 2011。

英文影印版由东南大学出版社出版 2012。此影印版的出版和销售得到出版权和销售权的所有者——O'Reilly Media, Inc. 的许可。

版权所有, 未得书面许可, 本书的任何部分和全部不得以任何形式重制。

深入浅出移动互联网 (影印版)

出版发行: 东南大学出版社

地 址: 南京四牌楼 2 号 邮编: 210096

出 版 人: 江建中

网 址: <http://www.seupress.com>

电子邮件: press@seupress.com

印 刷: 扬中市印刷有限公司

开 本: 787 毫米 × 980 毫米 12 开本

印 张: 39.5

字 数: 661 千字

版 次: 2012 年 6 月第 1 版

印 次: 2012 年 6 月第 1 次印刷

书 号: ISBN 978-7-5641-3414-3

定 价: 98.00 元 (册)

本社图书若有印装质量问题, 请直接与营销部联系。电话 (传真): 025-83791830

Table of Contents (Summary)

	Intro	xxi
1	Getting Started on the Mobile Web: <i>Responsive Web Design</i>	1
2	Responsible Responsiveness: <i>Mobile-first Responsive Web Design</i>	43
3	A Separate Mobile Website: <i>Facing less-than-awesome circumstances</i>	91
4	Deciding Whom to Support: <i>What devices should we support?</i>	137
5	Device Databases and Classes: <i>Get with the group</i>	151
6	Build a Mobile Web App Using a Framework: <i>The Tartanator</i>	217
7	Mobile Web Apps in the Real World: <i>Super mobile web apps</i>	267
8	Build Hybrid Mobile Apps with PhoneGap: <i>Tartan Hunt: Going native</i>	313
9	How to Be Future Friendly: <i>Make (some) sense of the chaos</i>	357
i	Leftovers: <i>The top six things (we didn't cover)</i>	373
ii	Set Up Your Web Server Environment: <i>Gotta start somewhere</i>	387
iii	Install WURFL: <i>Sniffing out devices</i>	397
iv	Install the Android SDK and Tools: <i>Take care of the environment</i>	403
	Index	417

Table of Contents (the real thing)

Intro

Your brain on mobile web. Here you are trying to learn something, while here your brain is, doing you a favor by making sure the learning doesn't stick. Your brain's thinking, "Better leave room for more important things, like which wild animals to avoid and whether setting this BlackBerry Bold on fire is going to activate the sprinkler system." So how do you trick your brain into thinking that your life depends on knowing mobile web?

Who is this book for?	xxii
We know what you're thinking	xxiii
And we know what your brain is thinking	xxiii
Metacognition: thinking about thinking	xxv
The technical review team	xxx
Acknowledgments	xxxix

getting started on the mobile web

1

Responsive Web Design

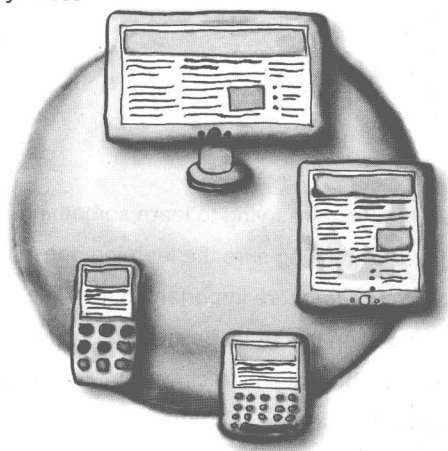
Hey there! Are you ready to jump into mobile?

Mobile web development is a wildly exciting way of life. There's glamour and excitement, and plenty of *Eureka!* moments. But there is also mystery and confusion. Mobile technology is evolving at bewildering speed, and there's so much to know! Hang tight. We'll start our journey by showing you a way of making websites called **Responsive Web Design (RWD)**. You'll be able to adapt websites to look great on a whole lot of mobile devices by building on the web skills you already have.

index.html



styles.css



Get on the mobile bandwagon	2
Something odd happened on the way to the pub	4
If mobile phone web browsers are so great...	5
What's so different about the mobile web?	6
Responsive Web Design	10
Different CSS in different places	12
CSS media queries	13
The current structure of the Splendid Walrus site	15
Analyze the current CSS	16
What needs to change?	17
Identify the CSS that needs to change	18
Steps to creating the mobile-specific CSS	19
What's wrong with a fixed-width layout, anyway?	26
How is fluid better?	27
The fluid formula	28
Continue your fluid conversion	29
Context switching	31
What's wrong with this picture?	32
Fluid images and media	33
Remember to be responsible	36
That's a responsive site!	40
Responsive design is also a state of mind	41

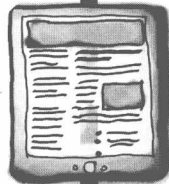
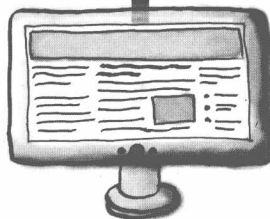
responsible responsiveness

2

Mobile-first Responsive Web Design**That's a beautiful mobile site. But beauty is only skin deep.**

Under the covers, it's a different thing entirely. It may look like a mobile site, but it's still a desktop site in mobile clothing. If we want this site to be greased lightning on mobile, we need to start with **mobile first**. We'll begin by dissecting the current site to find the desktop bones hiding in its mobile closet. We'll clean house and start fresh with **progressive enhancement**, building from the basic content all the way to a desktop view. When we're done, you'll have a page that is optimized regardless of the screen size.

Progressive enhancement based on screen size and client features

Very small screens
(feature phones)Small screens
(smartphones)Medium screens
(tablets)Larger screens
(desktops and TVs)

Just when you thought it was time to celebrate...	44
Is there really a problem? How do we know?	45
What to do when things aren't blazing fast	47
Don't let its looks fool you, that's a BIG page	48
There's gold in 'em HAR hills	49
Find the drags on page speed	51
Where did that Google Maps JavaScript come from?	53
It looks mobile friendly, but it isn't	55
Mobile-first Responsive Web Design	56
What is progressive enhancement?	57
Fix the content floats	60
Mobile-first media queries	61
Surprise! The page is broken in Internet Explorer	62
One src to rule them all	68
Zoom in on the viewport <meta> tag	72
The right to zoom?	73
Add the map back using JavaScript	74
Build a pseudo-media query in JavaScript	76
Add the JavaScript to the On Tap Now page	77
These widgets aren't responsive	79
Move iframe attributes to CSS equivalents	80
Remove attributes from the JavaScript	81
The map overlap is back	83
Let the content be your guide	84
Breakpoints to the rescue	87

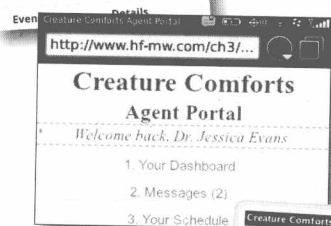
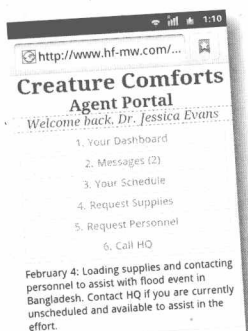
a separate mobile website

3

Facing less-than-awesome circumstances

The vision of a single, responsive Web is a beautiful one...

in which every site has one layout to rule them all, made lovingly with a mobile-first approach. Mmm...tasty. But what happens when a stinky dose of reality sets in? Like legacy systems, older devices, or customer budget constraints? What if, sometimes, instead of mixing desktop and mobile support into one lovely soup, you need to keep 'em separated? In this chapter, we look at the nitty-gritty bits of **detecting mobile users, supporting those cruffy older phones, and building a separate mobile site.**



Creature Comforts has agents in the field	92
How can agents get and share the info they need?	93
Send mobile users to a mobile-optimized website	96
Sniff out mobile users	97
Getting to know user agents	98
User agents: spawn of Satan?	101
Straight talk: Most major sites have a separate mobile website	104
When what you really want to do is (re-)direct	105
Take a peek at the script	106
How does the script work?	107
Make a mobile mockup	108
Special delivery...of complicating factors	110
Not all phones are smartphones...not by a sight	113
Let's keep it basic: Meet XHTML-MP	114
Why would we want to use that old thing?	115
Keep your nose clean with XHTML-MP	116
By the way, scrolling sucks	119
One last curveball	119
Access keys in action	123
What went wrong?	124
Fix the errors	125
Mobile-savvy CSS	127
Hmmm...something is missing	132
The button look is sorely missed!	133
Great success!	134

deciding whom to support

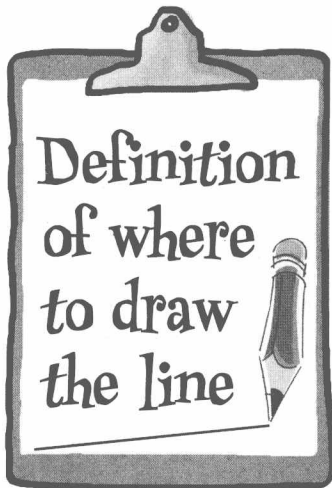
What devices should we support?

4

There aren't enough hours in the day to test on every device.

You have to draw the line somewhere on what you can support. **But how do you decide?** What about people using devices you can't test on—are they left out in the cold? Or is it possible to build your web pages in a way that will reach people on devices you've never heard of? In this chapter, we're going to mix a magic concoction of **project requirements** and **audience usage** to help us figure out **what devices we support** and **what to do about those we don't**.

How do you know where to draw the line?	138
Step away from the keyboard for a second	139
Things you <i>don't</i> support vs. those you <i>can't</i> support	140
Ask questions about your project	142
Ingredients for your magic mobile potion	144
Draw from your cupboard of tools and data	145
How do I know my customers have the right stuff?	150

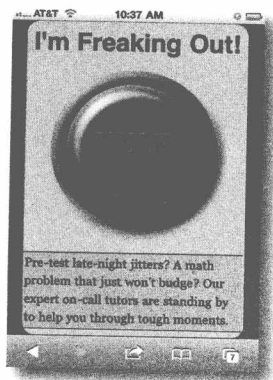
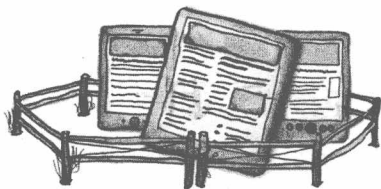


device databases and classes

5

Get with the group

Setting the bar for the devices we support doesn't take care of a few nagging issues. How do we find out enough stuff about our users' mobile browsers to know if they measure up before we deliver content to them? How do we avoid only building (lame) content for the lowest common denominator? And how do we organize all of this stuff so that we don't lose our minds? In this chapter, we'll enter the realm of **device capabilities**, learn to access them with a **device database**, and, finally, discover how to group them into **device classes** so that we can keep our sanity.



A panic button for freaked-out students	152
Mobile device data sources to the rescue	154
Meet WURFL	155
WURFL and its capabilities	156
WURFL: Clever API code	159
We can build an explore page, too	160
An explore page: Setting up our environment	161
A quick one-two punch to improve our explore page	168
Put capabilities to work	170
Use WURFL to help differentiate content	170
Initialize the device and get the info ready	172
Is this thing mobile?	172
Make the page a bit smarter with WURFL	176
The panic button: For phones only	177
Device classes	181
Expanding a lucrative part of AcedIt!'s business	182
Evaluate the home page wearing mobile-tinted glasses	183
Group requirements into multiple mobile flavors	184
Rounding out our device classes	185
Get acquainted with the matching function	191
What's going on in that switch statement?	192
Use the matching function to test capabilities	193
Fill in the gaps in the device class tests	200
We need a bigger safety net	211
A stitch in time	212

build a mobile web app using a framework

The Tartanator

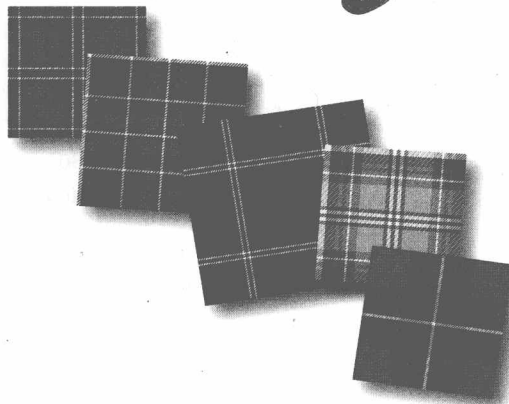
6

“We want an app!” Just a year or two ago, that hallmark cry generally meant one thing: native code development and deployment for each platform you wanted to support. But native isn’t the only game in town. These days, web-based apps for mobile browsers have some street cred—especially now that hip cat **HTML5** and his sidekicks, **CSS3** and **JavaScript**, are in the house. Let’s dip our toes into the mobile web app world by taking a **mobile framework**—code tools designed to help you get your job done quickly—for a spin!

Hmmm...it's...nice,
but can you make
it feel more...like a
native app?



HTML5...app...what do these words even mean?	219
How “traditional” websites typically behave	220
How applike websites often behave	221
The master plan for phase 1 of the Tartanator	224
Why use mobile web app frameworks?	225
Our choice for the Tartanator: jQuery Mobile	226
Build a basic page with jQuery Mobile	228
Mark up the rest of the page	229
The HTML5 data-* attribute	231
Link to multiple pages with jQuery Mobile	234
Take the list from blah to better	241
Drop in the rest of the tartans	243
Filter and organize a list	244
Add a footer toolbar	249
Make the toolbar snazzy	250
Finalize the structure	251
Time to make that tartan-building form	253
Translate tartan patterns to a form	255
Build an HTML5 form	256
It’s time to add some basic fields	257
Lists within lists let the users add colors	258
Color-size ingredient pairs: The color select field	259
Color-size field pairs: The size field	260
Link to the form	262



mobile web apps in the real world

7

Super mobile web apps

The mobile web feels like that gifted kid in the class.

You know, kind of fascinating, capable of amazing things, but also a mysterious, unpredictable troublemaker. We've tried to keep its hyperactive genius in check by being mindful of constraints and establishing boundaries, but now it's time to capitalize on some of the mobile web's natural talents.

We can use **progressive enhancement** to spruce up the interface in more precocious browsers and transform erratic connectivity from a burden to a feature by crafting a thoughtful **offline mode**. And we can get at the essence of mobility by using **geolocation**. Let's go make this a super mobile web app!



It looks nice...	268
Mobile apps in the real world	270
Ready, set, enhance!	274
Make a better form	275
A widget to manage the list of colors and sizes	276
A peek under the hood	277
So, that's the frontend enhancement...	278
...and now for the backend	280
The two sides of generate.php	281
One last thing!	282
Offline is important	284
A basic recipe to create a cache manifest	285
Dev tools to the rescue	286
Serve the manifest as the correct content-type	287
Victory is (finally) ours	297
How geolocation works	298
How to ask W3C-compliant browsers where they are	299
Start in on the Find Events page: The baseline	301
Let's integrate geolocation	303
Nothing found	309

build hybrid mobile apps with PhoneGap

8

Tartan Hunt: Going native

Sometimes you've got to go native. It might be because you need access to something not available in mobile browsers (yet). Or maybe your client simply *must* have an app in the App Store. We look forward to that shiny future when we have access to everything we want in the browser, and mobile web apps share that sparkly allure native apps enjoy. Until then, we have the option of **hybrid development**—we continue writing our **code using web standards**, and use a **library to bridge the gaps** between our code and the device's native capabilities. **Cross-platform native apps built from web technologies?** Not such a bad compromise, eh?

Opportunity knocks again	314
How do hybrid apps work?	317
Bridge the web-native gap with PhoneGap	318
Get acquainted with PhoneGap Build	321
How will the app work?	322
Keep track of discovered tartans	323
Anatomy of the Tartan Hunt project	324
Download your apps	328
Choose your adventure	329
Who's seen what? Store found tartans	334
What can localStorage do for us?	335
Check out what a browser supports	339
Use a function to show which tartans are found	340
The toggle and toggleClass methods	341
You found a tartan, eh? Prove it!	344
Rope in PhoneGap to take pictures	345
PhoneGap is almost ready for its close-up	347
Now we're ready for the mediaCapture API	348
How will we handle the success?	349
It always looks a bit different in real life	350
It's just a bit anonymous	351
One last thing!	353
We nailed it!	354

Hybrid App
Bridge



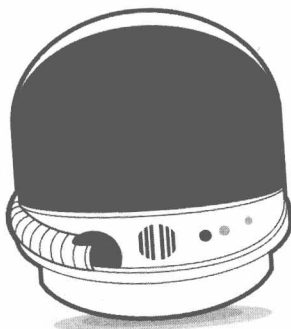
how to be future friendly

9

Make (some) sense of the chaos

Responsive Web Design. Device detection, Mobile web apps. PhoneGap. Wait...which one should we use?

There are an overwhelming number of ways to develop for the mobile web. Often, projects will involve **multiple techniques used in combination**. There is no single right answer. But don't worry. The key is to learn to go with the flow. **Embrace the uncertainty**. Adopt a **future-friendly mindset** and ride the wave, confident that you're flexible and ready to adapt to whatever the future holds.



Now what?	358
Time to dispel our collective illusions of control	361
A future-friendly manifesto	362
If you can't be future proof, be future friendly	364
App today, web page tomorrow	365
It's a long journey: Here are some guideposts	366
Mix up a batch of mobile goodness	369
Look toward the future	371



leftovers



The top six things (we didn't cover)

Ever feel like something's missing? We know what you mean... Just when you thought you were done, there's more.

We couldn't leave you without a few extra details, things we just couldn't fit into the rest of the book. At least, not if you want to be able to carry this book around without a metallic case and caster wheels on the bottom. So take a peek and see what you (still) might be missing out on.

#1. Testing on mobile devices	374
#2. Remote debugging	376
#3. Determine which browsers support what	382
#4. Device APIs	384
#5. Application stores and distribution	385
#6. RESS: REsponsive design + Server-Side components	386

set up your web server environment



Gotta start somewhere

You can't spell "mobile web" without the "web." There are no two ways about it. You're going to need a web server if you want to develop for the mobile web. That goes for more than just completing the exercises in this book. You need somewhere to put your web-hosted stuff, whether you use a third-party commercial web hosting service, an enterprise-class data center, or your own computer. In this appendix, we'll walk you through the steps of **setting up a local web server** on your computer and **getting PHP going** using free and open source software.

What we need from you	388
Only available locally	389
Windows and Linux: Install and configure XAMPP	390
Get going with XAMPP	391
Mac folks: It's MAMP time	392
Make sure you dock at the right port	393
Access your web server	394
phpInfo, please!	396



install WURFL

Sniffing out devices

The first step to solving device detection mysteries is a bit of legwork. Any decent gumshoe knows we've got to gather our clues and interrogate our witnesses. First, we need to seek out the brains of the operation: the **WURFL PHP API**. Then we'll go track down the brawn: capability information for thousands of devices in a single **XML data file**. But it'll take a bit of coaxing to get the two to spill the whole story, so we'll tweak a bit of **configuration** and take some careful notes.

Who's got the brains?	398
And who's got the brawn?	399
Getting the two to work together	400
A bit of filesystem housekeeping	401
Take note!	402

install the Android SDK and tools



Take care of the environment

To be the master of testing native Android apps, you need to be environmentally aware. You'll need to turn your computer into a nice little ecosystem where you can herd Android apps to and from virtual (emulated) or real devices. To make you the shepherd of your Android sheep, we'll show you how to download the **Android software development kit (SDK)**, how to install some **platform tools**, how to **create some virtual devices**, and how to **install and uninstall apps**.

Let's download the Android SDK	404
Get the right tools for the job	405
Create a new virtual device	408
Find the right PATH	413

Index

417

1 getting started on the mobile web

* Responsive Web Design *



Dashing, exciting, fascinating,
and oh-so-popular...but am I
ready to take the plunge?

Hey there! Are you ready to jump into mobile?

Mobile web development is a wildly exciting way of life. There's glamour and excitement, and plenty of *Eureka!* moments. But there is also mystery and confusion. Mobile technology is evolving at bewildering speed, and there's so much to know! Hang tight. We'll start our journey by showing you a way of making websites called **Responsive Web Design (RWD)**. You'll be able to adapt websites to look great on a whole lot of mobile devices by building on the web skills you already have.

Get on the mobile bandwagon

There's a pretty good chance you own a mobile phone. We know that not simply because you bought this book (smart move, by the way!), but because it's hard to find someone who doesn't own a mobile phone.

It doesn't matter where you go in the world. Mobile phones are being used everywhere, from farmers in Nigeria using their mobiles to find which market has the best price for their crops, to half of Japan's top 10 best-selling novels being consumed and written—*yes, written*—on mobile phones.

At the beginning of 2011, there were 5.2 billion phones being used by the 6.9 billion people on Earth. **More people use mobile phones than have working toilets or toothbrushes.**

The time is now

So yeah, mobile is huge, but it's been big for years. Why should you get on the mobile bandwagon now?

Because **the iPhone changed everything**. It sounds clichéd, but it is true. There were app stores, touchscreens, and web browsers on phones before the iPhone, but Apple was the first to put them together in a way that made it easy for people to understand and use.

