LEARNING MADE EASY

In Full Color

Coding

for dummies®

A Wiley Brand

Practice coding online at codecademy

Learn the basics of five different coding languages

Apply coding skills to build an application

Nikhil Abraham

Coding

for dummies®

A Wiley Brand

Coding

for dummies®

A Wiley Brand

by Nikhil Abraham

for dummies®
A Wiley Brand

Coding For Dummies®

Published by: **John Wiley & Sons, Inc.,** 111 River Street, Hoboken, NJ 07030-5774, www.wiley.com

Copyright © 2015 by John Wiley & Sons, Inc., Hoboken, New Jersey

Media and software compilation copyright © 2015 by John Wiley & Sons, Inc. All rights reserved.

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the Publisher. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at www.wiley.com/go/permissions.

Trademarks: Wiley, For Dummies, the Dummies Man logo, Dummies.com, Making Everything Easier, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and may not be used without written permission. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002. For technical support, please visit www.wiley.com/techsupport.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at http://booksupport.wiley.com. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2014954659

ISBN 978-1-119-29332-3 (pbk); ISBN 978-1-119-29610-2 (ebk); ISBN 978-1-119-29607-2 (ebk)

Coding For Dummies (9781119293323) was previously published as Coding For Dummies (9781118951309). While this version features a new Dummies cover and design, the content is the same as the prior release and should not be considered a new or updated product.

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Contents at a Glance

Table of Contents

Introduction

The ability to read, write, and understand code has never been more important, useful, or lucrative as it is today. Computer code has forever changed our lives. Some people can't even make it through the day without interacting with something built with code. Even so, for many people, the world of coding seems complex and inaccessible. Maybe you participated in a tech-related business meeting and did not fully understand the conversation. Perhaps you tried to build a web page for your family and friends, but ran into problems displaying pictures or aligning text. Maybe you're even intimidated by the unrecognizable words on the covers of books about coding: words such as HTML, CSS, JavaScript, Python, or Ruby.

If you've previously been in these situations, then *Coding For Dummies* is for you. This book explains basic concepts so you can participate in technical conversations, and ask the right questions. Don't worry — in this book I've assumed you are starting with little to no previous coding knowledge, and I haven't tried to cram every possible coding concept into these pages. Additionally, I encourage you here to learn by doing, and by actually creating your own programs. Instead of a website, imagine that you wanted to build a house. You could spend eight years studying to be an architect, or you could start today by learning a little bit about foundations and framing. This book kickstarts your coding journey today.

The importance of coding is ever increasing. As author and technologist Douglas Rushkoff famously said, "program or be programmed." When humans invented languages and then the alphabet, people learned to listen and speak, and then read and write. In our increasingly digital world, it is important to learn not just how to use programs, but how to make them as well. For example, observe this transition in music. For over a century, music labels decided what songs the public could listen to and purchase. In 2005, three coders created YouTube, which allowed anyone to release songs. Today more songs have been uploaded to YouTube than have been released by all the record labels in the last century combined.

Accompanying this book are examples at `www.codecademy.com`, whose exercises are one of the easiest ways to learn how to code without installing or downloading anything. The Codecademy companion site includes examples and exercises from this book, along with projects and examples for additional practice.

About This Book

This book is designed for readers with little to no coding experience, and gives an overview of programming to non-programmers. In plain English, you learn how code is used to create web programs, who makes those programs, and the processes they use. The topics covered include:

- Explaining what coding is and answering the common questions related to code.
- Building basic websites using the three most common languages: HTML, CSS, and JavaScript.
- Surveying other programming languages such as Ruby and Python.
- Building an application using everything you learn in the book.

As you read this book, keep the following in mind:

- The book can be read from beginning to end, but feel free to skip around if you like. If any topic interests you, start there. You can always return to the previous chapter, if necessary.
- At some point you will get stuck, and code you write will not work as intended. Do not fear! There are many resources to help you including support forums, others on the Internet, and me! Using Twitter, you can send me a public message at `@nikhilgabraham` with the hashtag `#codingFD`.
- Code in the book will appear in a monospaced font like this: `<h1>Hi there! </h1>`.

Foolish Assumptions

I do not make many assumptions about you, the reader, but I do make a few:

I assume you don't have previous programming experience. To follow along, then, you only need to be able to read, type, and follow directions. I try to explain as many concepts as possible using examples and analogies you already know.

I assume you have a computer running the latest version of Google Chrome. The examples in the book have been tested and optimized for the Chrome browser, which is available for free from Google. Even so, the examples may also work in

the latest version of Firefox. Using Internet Explorer for the examples in this book, however, is discouraged.

I assume you have access to an Internet connection. Some of the examples in the book can be done without an Internet connection, but most require one so you can access and complete the exercises on `www.codecademy.com`.

Icons Used in This Book

Here are the icons used in the book to flag text that should be given extra attention or can be skipped.

This icon flags useful information or explains a shortcut to help you understand a concept.

This icon explains technical details about the concept being explained. The details might be informative or interesting, but are not essential to your understanding of the concept at this stage.

Try not to forget the material marked with this icon. It signals an important concept or process that you should keep in mind.

Watch out! This icon flags common mistakes and problems that can be avoided if you heed the warning.

Beyond the Book

A lot of extra content that you won't find in this book is available at `www.dummies.com`. Go online to find the following:

- **The source code for the examples in this book and a link to the Codecademy exercises:** You can find these at

  ```
  www.dummies.com/go/codingfd
  ```

  The source code is organized by chapter. The best way to work with a chapter is to download all the source code for it at one time.

- **Cheat Sheet:** You can find a list of common HTML, CSS, and JavaScript commands, among other useful information, at

  To view this book's Cheat Sheet, simply go to `www.dummies.com` and search for "Coding For Dummies Cheat Sheet" in the Search box.

- **Extras:** Additional articles with extra content are posted for roughly each section of the book. You can access these additional materials at

  ```
  www.dummies.com/extras/coding
  ```

- **Updates:** Code and specifications are constantly changing, so the commands and syntax that work today may not work tomorrow. You can find any updates or corrections by visiting

  ```
  www.dummies.com/extras/coding
  ```

Where to Go from Here

All right, now that all of the administrative stuff is out of the way, it's time to get started. You can totally do this. Congratulations on taking your first step into the world of coding!

1 Getting Started with Coding

IN THIS PART . . .

Understand what code is and what you can build with it.

Review programming languages used to write code.

Code for the web using front-end and back-end programming languages.

Follow the process programmers use to create code.

Write your first program using code.

IN THIS CHAPTER

- Seeing what code is and what it can do
- Touring your first program using code
- Understanding programming languages used to write code

Chapter 1
What Is Coding?

"A million dollars isn't cool, you know what's cool? A billion dollars."
— SEAN PARKER, THE SOCIAL NETWORK

Every week the newspapers report on another technology company that has raised capital or sold for millions of dollars. Sometimes, in the case of companies like Instagram, WhatsApp, and Uber, the amount in the headline is for billions of dollars. These articles may pique your curiosity, and you may want to see how code is used to build the applications that experience these financial outcomes. Alternatively, your interests may lie closer to work. Perhaps you work in an industry in decline, like print media, or in a function that technology is rapidly changing, like marketing. Whether you are thinking about switching to a new career or improving your current career, understanding computer programming or "coding" can help with your professional development. Finally, your interest may be more personal — perhaps you have an idea, a burning desire to create something, a website or an app, to solve a problem you have experienced, and you know reading and writing code is the first step to building your solution. Whatever your motivation, this book will shed light on coding and programmers, and help you think of both not as mysterious and complex but approachable and something you can do yourself.

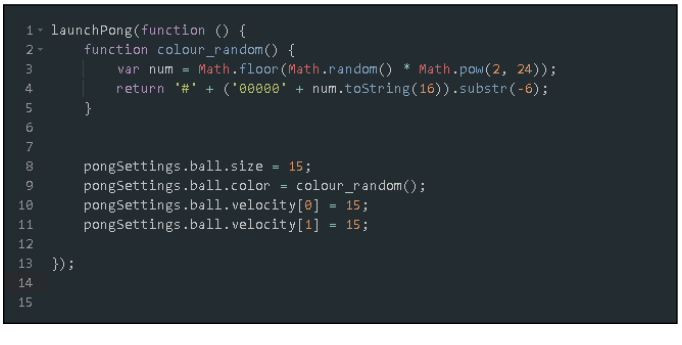
In this chapter, you will understand what code is, what industries are affected by computer software, the different types of programming languages used to write code, and take a tour of a web app built with code.

Defining What Code Is

Computer code is not a cryptic activity reserved for geniuses and oracles. In fact, in a few minutes you will be writing some computer code yourself! Most computer code performs a range of tasks in our lives from the mundane to the extraordinary. Code runs our traffic lights and pedestrian signals, the elevators in our buildings, the cell phone towers that transmit our phone signals, and the space ships headed for outer space. We also interact with code on a more personal level, on our phones and computers, and usually to check email or the weather.

Following instructions

Computer code is a set of statements, like sentences in English, and each statement directs the computer to perform a single step or instruction. Each of these steps is very precise, and followed to the letter. For example, if you are in a restaurant and ask a waiter to direct you to the restroom, he might say, "head to the back, and try the middle door." To a computer, these directions are so vague as to be unusable. Instead, if the waiter gave instructions to you as if you were a computer program he might say, "From this table, walk northeast for 40 paces. Then turn right 90 degrees, walk 5 paces, turn left 90 degrees, and walk 5 paces. Open the door directly in front of you, and enter the restroom." Figure 1-1 shows lines of code from the popular game, Pong. Do not worry about trying to understand what every single line does, or feel intimated. You will soon be reading and writing your own code.



```
launchPong(function () {
    function colour_random() {
        var num = Math.floor(Math.random() * Math.pow(2, 24));
        return '#' + ('00000' + num.toString(16)).substr(-6);
    }


    pongSettings.ball.size = 15;
    pongSettings.ball.color = colour_random();
    pongSettings.ball.velocity[0] = 15;
    pongSettings.ball.velocity[1] = 15;

});
```

FIGURE 1-1: Computer code from the game Pong.

One rough way to measure a program's complexity is to count its statements or lines of code. Basic applications like the Pong game have 5,000 lines of code, while more complex applications like Facebook currently have over 10 million lines of code. Whether few or many lines of code, the computer follows each instruction exactly and effortlessly, never tiring like the waiter might when asked for the 100th time for the location of the restroom.

TIP

Be careful of only using lines of code as a measure for a program's complexity. Just like when writing in English, 100 well written lines of code can perform the same functionality as 1,000 poorly written lines of code.

Writing code with some Angry Birds

If you have never written code before, now is your chance to try! Go to `http://csedweek.org/learn` and under the heading "Tutorials for Beginners" click the "Write Your First Computer Program" link with the Angry Birds icon, as shown in Figure 1-2. This tutorial is meant for those with no previous computer programming experience, and introduces the basic building blocks used by all computer programs. The most important take-away from the tutorial is to understand that computer programs use code to literally and exactly tell the computer to execute a set of instructions.

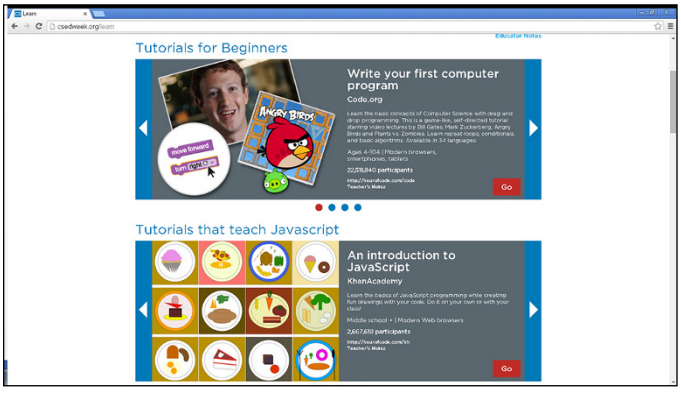


FIGURE 1-2: Write your first computer program with a game-like tutorial using Angry Birds.

TIP

Computer Science Education Week is an annual program dedicated to elevating the profile of computer science during one week in December. In the past, President Obama, Bill Gates, basketball player Chris Bosh, and singer Shakira, among others, have supported and encouraged people from the US and around the world to participate.

Understanding What Coding Can Do for You

Coding can be used to perform tasks and solve problems that you experience every day. The "everyday" situations in which programs or apps can provide assistance continues to grow at an exponential pace, but this was not always the case. The

rise of web applications, internet connectivity, and mobile phones have inserted software programs into daily life, and lowered the barrier for you to become a creator, solving personal and professional problems with code.

Eating the world with software

In 2011, Marc Andreessen, creator of Netscape Navigator and now venture capitalist, noted that "software is eating the world." He predicted that software companies would disrupt existing companies at a rapid pace. Traditionally, code powered software used on desktops and laptops. The software had to first be installed, and then you had to supply data to the program. Three trends have dramatically increased the use of code in everyday life:

- **Web-based software:** This software operates in the browser without requiring installation. For example, if you wanted to check your email, you previously had to install an email client either by downloading the software or from a CD-ROM. Sometimes, issues arose when the software was not available for your operating system, or conflicted with your operating system version. Hotmail, a web-based email client, rose to popularity, in part, because it allowed users visiting `www.hotmail.com` to instantly check their email without worrying about installation or software compatibility. Web applications increased consumer appetite to try more applications, and developers in turn were incentivized to write more applications.
- **Internet broadband connectivity:** Broadband connectivity has increased, providing a fast Internet connection to more people in the last few years than in the previous decade. Today, more than two billion people can access web-based software, up from approximately 50 million only a decade ago.
- **Mobile phones:** Today's smartphones bring programs with you wherever you go, and help supply data to programs. Many software programs became more useful when accessed on-the-go than when limited to a desktop computer. For instance, use of maps applications greatly increased thanks to mobile phones because users need directions the most when lost, not just when planning a trip at home on the computer. In addition, mobile phones are equipped with sensors that measure and supply data to programs like orientation, acceleration, and current location through GPS. Now instead of having to input all the data to programs yourself, mobile devices can help. For instance, a fitness application like RunKeeper does not require you to input start and end times to keep track of your runs. You can press start at the beginning of your run, and the phone will automatically track your distance, speed, and time.

The combination of these trends have created software companies that have upended incumbents in almost every industry, especially ones typically immune to technology. Some notable examples include:

- **Airbnb:** Airbnb is a peer-to-peer lodging company that owns no rooms, yet books more nights than the Hilton and Intercontinental, the largest hotel chains in the world. (See Figure 1-3.)
- **Uber:** Uber is a car transportation company that owns no vehicles, books more trips, and has more drivers in the largest 200 cities than any other car or taxi service.
- **Groupon:** Groupon, the daily deals company, generated almost $1 billion after just two years in business, growing faster than any other company in history, let alone any other traditional direct marketing company.

FIGURE 1-3
Airbnb booked 5 million nights after 3.5 years, and its next 5 million nights 6 months later.

Coding on the job

Coding can be useful in the workplace as well. Outside the technology sector, coding in the workplace is common for some professions like financial traders, economists, and scientists. However, for most professionals outside the technology sector, coding is just beginning to penetrate the workplace, and gradually starting to increase in relevance. Here are areas where coding is playing a larger role on the job:

- **Advertising:** Spend is shifting from print and TV to digital campaigns, and search engine advertising and optimization relies on keywords to bring visitors to websites. Advertisers who understand code see successful keywords used by competitors, and use that data to create more effective campaigns.
- **Marketing:** When promoting products, personalizing communication is one strategy that often increases results. Marketers who code can query customer databases and create personalized communications that include customer names and products tailored to specific interests.
- **Sales:** The sales process always starts with leads. Salespeople who code retrieve their own leads from web pages and directories, and then sort and quality those leads.


TIP

Retrieving information by copying text on web pages and in directories is referred to as *scraping*.

- **Design:** After creating a web page or a digital design, designers must persuade other designers and eventually developers to actually program their drawings into the product. Designers who code can more easily bring their designs to life, and can more effectively advocate for specific designs by creating working prototypes that others can interact with.
- **Public relations:** Companies constantly measure how customers and the public react to announcements and news. For instance, if a celebrity spokesperson for a company does or says something offensive, should the company dump the celebrity? Public relations people who code query social media networks like Twitter or Facebook, and analyze hundreds of thousands of individual messages to understand market sentiment.
- **Operations:** Additional profit can be generated, in part, by analyzing a company's costs. Operations people who code write programs to try millions of combinations to optimize packaging methods, loading routines, and delivery routes.

Scratching your own itch (and becoming rich and famous)

Using code built by others and coding in the workplace may cause you to think of problems you personally face that you could solve with code of your own. You may have an idea for a social network website, a better fitness app, or something new altogether. The path from idea to functioning prototype used by others involves a good amount of time and work, but might be more achievable than you think. For example, take Coffitivity, a productivity website that streams ambient coffee shop sounds to create white noise. The website was created by two people who had just learned how to program a few months prior. Shortly after Coffitivity launched, Time Magazine named the website one of 50 Best Websites of 2013, and the Wall

Street Journal also reviewed the website. While not every startup or app will initially receive this much media coverage, it can be helpful to know what is possible when a solution really solves a problem.

Having a goal, like a website or app you want to build, is one of the best ways to learn how to code. When facing a difficult bug or a hard concept, the idea of bringing your website to life will provide the motivation you need to keep going. Just as important, do not learn how to code to become rich and famous, as the probability of your website or app becoming successful is largely due to factors out of your control.


TIP

The characteristics that make a website or app addictive are described using the Hook Model here `http://techcrunch.com/2012/03/04/how-to-manufacture-desire`. Products are usually made by companies, and the characteristics of an enduring company are described here `http://www.sequoiacap.com/grove/posts/ya16/elements-of-enduring-companies`, based on a review of companies funded by Sequoia, one of the most successful venture capital firms in the world and early investors in Apple, Google, and PayPal.

Surveying the Types of Programming Languages

Code comes in different flavors called *programming languages.* Some popular programing languages are shown in Figure 1-4.

FIGURE 1-4: Some popular programming languages.

You can think of programming languages just like spoken languages, as they both share many of the same characteristics, such as:

- **Functionality across languages:** Programming languages can all create the same functionality similar to how spoken languages can all express the same objects, phrases, and emotions.
- **Syntax and structure:** Commands in programming languages can overlap just like words in spoken languages overlap. To output text to screen in Python or Ruby you use the `print` command, just like imprimer and imprimir are the verbs for "print" in French and Spanish.
- **Natural lifespan:** Programming languages are born when a programmer thinks of a new or easier way to express a computational concept. If other programmers agree, they adopt the language for their own programs and the programming language spreads. However, just like Latin or Aramaic, if the programming language is not adopted by other programmers or a better language comes along, then the programming language slowly dies from lack of use.

Despite these similarities, programming languages also differ from spoken languages in a few key ways:

- **One creator:** Unlike spoken languages, programming languages can be created by one person in a short period of time, sometimes in just a few days. Popular languages with a single creator include JavaScript (Brendan Eich), Python (Guido van Rossum), and Ruby (Yukihiro Matsumoto).
- **Written in English:** Unlike spoken languages (except, of course, English), almost all programming languages are written in English. Whether they're programming in HTML, JavaScript, Python, or Ruby, Brazilian, French, or Chinese programmers all use the same English keywords and syntax in their code. Some non-English programming languages exist, such as languages in Hindi or Arabic, but none of these languages are widespread or mainstream.

Comparing low-level and high-level programming languages

One way to classify programming languages is either as low-level languages or high-level languages. Low-level languages interact directly with the computer processor or CPU, are capable of performing very basic commands, and are generally hard to read. Machine code, one example of a low-level language, uses code that consists of just two numbers — 0 and 1. Figure 1-5 shows an example of

machine code. Assembly language, another low-level language, uses keywords to perform basic commands like read data, move data, and store data.

FIGURE 1-5: Machine code consists of 0s and 1s.

By contrast, high-level languages use natural language so it is easier for people to read and write. Once code is written in a high-level language, like C++, Python, or Ruby, an interpreter or compiler translates this high-level language into low-level code a computer can understand.

Contrasting compiled code and interpreted code

High-level programming languages must be converted to low-level programming languages using an interpreter or compiler, depending on the language. Interpreted languages are considered more portable than compiled languages, while compiled languages execute faster than interpreted languages. However, the speed advantage compiled languages have is starting to fade in importance as improving processor speeds make performance differences between interpreted and compiled languages negligible.

High-level programming languages like JavaScript, Python, and Ruby are interpreted. For these languages the interpreter executes the program directly, translating each statement *one line at a time* into machine code. High-level programming languages like C++, COBOL, and Visual Basic are compiled. For these languages, after the code is written a compiler translates *all* the code into machine code, and an executable file is created. This executable file is then distributed via the internet, CD-ROMs, or other media and run. Software you install on your computer, like Microsoft Windows or Mac OS X, are coded using compiled languages, usually C or C++.

Programming for the web

Software accessible on websites is gradually starting to take over installed software. Think of the last time you downloaded and installed software for your computer — you may not even remember! Installed software like Windows Media Player and Winamp that play music and movies have been replaced with websites like YouTube and Netflix. Traditional installed word processor and spreadsheet software like Microsoft Word and Excel are starting to see competition from web software like Google Docs and Sheets. Google is even selling laptops called Chromebooks that contain no installed software, and instead rely exclusively on web software to provide functionality.

The remainder of this book will focus on developing and creating web software, not just because web software is growing rapidly, but also because programs for the web are easier to learn and launch than traditional installed software.

Taking a Tour of a Web App Built with Code

With all this talk of programming, let us actually take a look at a web application built with code. `Yelp.com` is a website that allows you to search and find crowdsourced reviews for local businesses like restaurants, nightlife, and shopping. As shown in Figure 1-6, Yelp did not always look as polished as it does today, but its purpose has stayed relatively constant over the years.

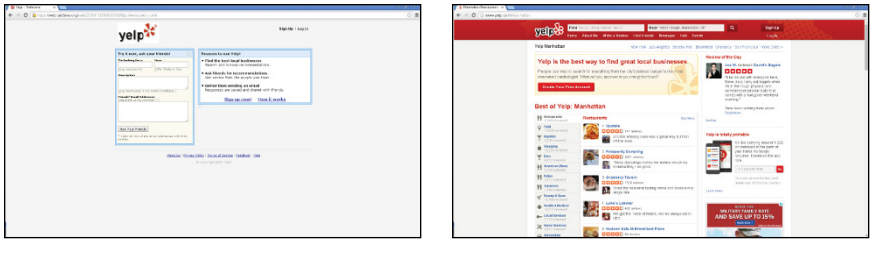


FIGURE 1-6: Yelp's website in 2004 and in 2014.

Defining the app's purpose and scope

Once you understand an app's purpose, you can identify a few actionable tasks a user should be able to perform to achieve that purpose. Regardless of design, the Yelp's website has always allowed users to