

# A Web-Based Introduction to Programming

Essential Algorithms, Syntax and Control  
Structures Using PHP and XHTML

Mike O'Kane

Copyright © 2008 Mike O'Kane

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, recording or otherwise, without the prior written permission of the author.

Please note: The information in this book is provided for instructional value and distributed on an "as is" basis, without warranty. While every precaution has been taken in the preparation of the book, neither the author nor Carolina Academic Press shall have any liability to any person or entity with respect to any loss or damage caused by or alleged to be caused, directly or indirectly, by the instructions contained in this book or by the programs or applications that are listed in, or provided as supplements to, this book.

Macintosh® and Mac OS® are registered trademarks of Apple, Inc. in the United States and other countries. Windows® is a registered trademark of Microsoft Corporation in the United States and other countries. Linux® is the registered trademark of Linus Torvalds in the United States and other countries. Mozilla® and Firefox® are registered trademarks of the Mozilla Foundation. Apache® is a trademark of the Apache Software Foundation. All product names identified in this book are trademarks or registered trademarks, and are the properties of their respective companies. We have used these names in an editorial fashion only, and to the benefit of the owner, with no intention of infringing the trademark.

Printed in the United States of America.

# Preface

This book is written in the hope of providing a measured, engaging, and carefully structured learning environment for anyone wishing to learn how to program. The book content has been field-tested in both traditional and online courses over an 18-month period by different instructors. Feedback from instructors and students alike has been overwhelmingly positive.

## ***Intended Audience***

The book is designed to serve:

- Instructors teaching introductory programming, programming logic and design, or Web programming courses, who want an effective textbook that engages students and provides a solid preparation for subsequent courses, but avoids overwhelming beginners with too much syntactical detail or program complexity.
- Traditional and online students taking a first course in programming, programming logic and design, or Web programming.
- Web designers, graphic artists, technical communicators, and others who find that their work increasingly requires some degree of programming expertise, and need an effective, hands-on introduction.
- Others who wish to learn the basics of programming, either for personal interest, or to explore the possibility of a career in this field.

## ***Approach***

The book takes a fairly novel approach, allowing students to learn program logic and design by developing a large number of small Web-based applications. Students love working with the Web, and this approach has other important benefits:

- Important concepts such as client/server design, server-side processing, and interface-driven code modules can be introduced in the form of working applications, and then applied in hands-on exercises.
- Students not only learn the essential control structures and syntax of a programming language, but also learn to use a markup language (and style sheets). This makes sense in today's programming environment where markup and programming are increasingly integrated components of a networked application.
- The material is relevant to students across a range of disciplines: Computer Science, Information Systems, Technical Communications, Network Systems, Digital Media, Web Technologies, Database Programming, and other technology-related fields.

- The focus on hands-on problem-solving and fundamental structures prepare students for next-level, language-specific courses such as Java or C++, without replicating a great deal of material, while the syntax covered here is generally consistent with these and other languages.

The book makes use of a programming language (PHP), and a scripting language (HTML), but does not attempt to provide a complete overview of either. Instead, students learn sufficient syntax to convert requirements into working applications using basic programming structures, arithmetic and logical expressions, user interfaces, functions, and data files. The focus remains on basic concepts, logic and design, algorithm development, and common programming procedures. The book provides context throughout, explaining why each topic is important, and referring students to related career paths.

Although the book focuses on Web-based applications, there is NO requirement for a network-based programming environment. The book uses a standalone Apache Web server (the open source xampp distribution provided by the Apache Friends group) that students can install on a USB drive or home computer simply by unzipping a file. As Chapter 2 demonstrates, students can begin programming in HTML and PHP in literally minutes.

## ***Features***

Each chapter begins with clearly stated learning outcomes. Each topic is introduced using examples of simple program requirements that are first developed as algorithms and interfaces and then realized in working code. Code statements and control structures are explained step by step.

Different programming topics are treated in separate chapters. Even topics that are commonly combined, such as counting loops and event-controlled loops, have their own chapters so that students have the chance to develop and apply their understanding of each separately.

Each chapter includes quizzes that have been carefully developed to test the student's understanding of the chapter's learning outcomes. The questions have been tested extensively in the classroom.

Three different types of coding exercise are provided at the end of each chapter:

- **Fixit exercises** provide small programs that include a single error of some kind. These exercises help students improve their problem-solving ability, test their understanding of key concepts, and develop tracing and debugging skills.
- **Modify exercises** provide working programs that must be modified to perform a somewhat different or additional function. These exercises help students determine how and where to add new code, and test their ability to read and understand existing code.
- **Code completion exercises** allow students to apply concepts and tools covered in the chapter by developing new applications. These exercises test the student's ability to: understand requirements, develop algorithms, and

produce working code. The code completion exercises follow consistent themes that are developed throughout the book, so that students can more readily appreciate the value of new functionalities that they learn in each chapter.

Templates for each exercise contain partially completed code so students don't waste time typing (and debugging) code that is not relevant to the problem at hand. The templates also help instructors to streamline the grading process.

The textbook CD includes a standalone Web server that can be installed on a fixed or portable drive simply by unzipping a file (so students can bring the software with them to work on computers at any location).

The server installation includes textbook containing all code samples and exercise templates. Students can complete the exercises simply by opening, editing, and saving the appropriate files. Assignments can be turned in simply by zipping and submitting the appropriate chapter folder.

The textbook appendices provide additional learning resources designed to: (a) help individual students with particular needs or interests (for example file/folder management, additional references, and help debugging code); and (b) deliver useful topics not included in the chapters (for example data representation, additional control structures, and multi-dimensional arrays).

The textbook Web site ensures that both students and instructors have access to the most current resources associated with this textbook. The Web site includes all materials found on the CD, and also provides access to additional exercises, test banks, slide presentations, quiz solutions, code solutions, and other instructional resources. The web site can be found at:

**<http://www.mikeokane.com/textbooks/WebTech/>**

## ***Chapter Overview***

**Chapter 1: Introducing Computer Programming.** Students learn the relationship between machine language and high-level languages, and review common tasks that computer programs typically perform. The work of a programmer is described, and the software development cycle is explained. The chapter highlights and summarizes significant important design approaches such as algorithm development, interface design, client/server design and object oriented programming. Different programming languages are identified, and the distinction is made between interpreted and compiled languages, and between markup and programming languages. Standalone and network applications are also contrasted.

**Chapter 2: Client/Server Applications – Getting Started.** This chapter prepares students for the hands-on work they will perform in subsequent chapters. File types and local and Internet addressing schemes are explained. Step-by-step instructions are provided to install, run, and test the required software. Students are then shown how to create, store, and run a number

of sample applications in order to become familiar with the process of using a text editor, saving files, running the Web server, and viewing the results in a Web browser.

**Chapter 3: Program Design – from Requirements to Algorithms.** The general characteristics and requirements of effective instructions are explored, using human and program examples. Students walk through the process of reviewing simple requirements, creating input, processing, and output (IPO) charts, designing the interface, and developing solution algorithms. The chapter introduces sequence, selection and control structures, variables and assignment operations, and arithmetic and logical expressions.

**Chapter 4: Basics of Markup – Creating a User Interface in HTML.** This chapter explains the significance of data rendering, and provides a brief overview and history of Hypertext Markup Language (HTML), up to the present XHTML implementation. Commonly used HTML tags are explained, and the student is shown how to apply these to create and organize simple Web pages. Cascading style sheets are introduced. Students are shown how to create HTML forms to obtain user input as a first step in developing interactive Web applications.

**Chapter 5: Creating a Working Program - Basics of PHP.** This chapter teaches sufficient PHP language syntax to process user input received from HTML forms, perform simple arithmetic, and produce formatted output. In the process, students learn to code arithmetic expressions, use standard operators and functions, create and work with variables, and identify and fix both syntax and logical errors.

**Chapter 6: Persistent Data – Working with Files and Databases.** This chapter explains the difference between persistent and transient data, and introduces text file processing as well as basic database concepts. Students learn to: open, read, write, and close text files; work with multiple files; parse lines of data that contain multiple values separated by some kind of delimiter.

**Chapter 7: Programs that Choose – Introducing Selection Structures.** This chapter introduces selection control structures and demonstrates the use of algorithms to solve problems requiring simple selection. Students learn to use IF and IF..ESE structures, Boolean expressions, relational operators, truth tables, simple string comparisons, and testing procedures.

**Chapter 8: Multiple Selection, Nesting, AND's and OR's.** This chapter develops examples from Chapter 7 to handle problems associated with input validation and more complex requirements. Students explore the use of compound Boolean expressions, nested selection structures, chained IF..ELSEIF..ELSE selection structures, and multiple but independent selection structures.

**Chapter 9: Programs that Count – Harnessing the Power of Repetition.** This chapter introduces loop structures with a focus on count-controlled FOR loops. Students learn how to refer to the counting variable within the

loop, and how to use loops to generate tables, crunch numbers, accumulate totals, find highest and lowest values in a series, select values from a file of records, and display bar charts.

**Chapter 10: "While NOT End-Of-File" – Introducing Event-Controlled Loops.** This chapter introduces WHILE loops and demonstrates the use of the priming read and the standard algorithm to process files of unknown length. The student is shown how WHILE loops can be used to perform various operations on a list of data values, and how a file of records can be processed and searched for specific records or field values.

**Chapter 11: Structured Data – Working with Arrays.** This chapter introduces numerically-indexed and associative arrays, and shows how arrays can be used to store, access, and update multiple-related values. The use of the FOR loop to process arrays is explained, and various array-processing algorithms are demonstrated. Students learn how to use associative arrays as lookups, and gain a better understanding of the way that data is received from HTML forms. Web sessions are introduced, and students learn how to use session variables to maintain session data between applications.

**Chapter 12: Program Modularity – Working with Functions and Objects.** This chapter demonstrates the importance of program modularity and introduces functions, include files and objects. Students learn to write their own functions, to build libraries of related functions, and to call functions from different applications as needed. Key concepts and examples of object oriented programming are also introduced in this chapter.

**Chapter 13: In Conclusion..** This last chapter provides a short overview of key concepts and technologies that the students may want to explore after completing this textbook.

The textbook also includes a number of useful appendices as follows:

**Appendix A** introduces data representation, and shows how binary values can store data for a wide range of purposes.

**Appendix B** provides an introduction to overview of file and folder management, file addressing schemes (including relative and absolute addresses), and the use of the command line with a list of common DOS and Unix command equivalents.

**Appendix C** provides instructions to install the Web server under Windows, Mac OS X, and Linux.

**Appendix D** provides extensive debugging help for students having trouble identifying and resolving code errors.

**Appendix E** provides additional material and references for students wishing to learn more about HTML and style sheets.

**Appendix F** provides additional material and references for students wishing to learn more about PHP.

**Appendix G** covers topics that were intentionally left out of the chapters to avoid overwhelming the beginning student (for example, shortcut operators, the SWITCH statement, DO..WHILE loops, and multi-dimensional arrays).

# Acknowledgments

This textbook could not have been created without the generous help and support of many others. In particular I want to thank my dear wife Constance Humphries for her invaluable technical advice, proof-reading, and daily encouragement and patience! My sincere thanks to Bob Conroy, Emily Utt, and Tim Colton at Carolina Academic Press for their supportive style, professionalism and experience. Thanks to my fellow instructors at Asheville-Buncombe Technical Community College, especially to Charlie Wallin and Fred Smartt who field-tested the book, made invaluable suggestions and submitted any number of corrections. And thanks to all of those students who have learned with me and sometimes in spite of me as this book evolved in the classroom.

And a huge thank you to Kai 'Oswald' Seidler, Kay Vogelgesang, and all those who have contributed to the Apache Friends Project, and who continue to deliver and support the XAMPP distribution. So many of us owe you our great appreciation for your generosity of spirit!

# About the Author

Mike O'Kane holds a master's degree in Systems Science (specializing in Advanced Technology) from Binghamton University. He has eight years experience teaching computer science courses and currently teaches at Asheville Buncombe Technical Community College in North Carolina. He also has extensive practical experience in the use of technology for learning, having worked at IBM as a short-course developer, NC State University as an Instructional Coordinator, and the University of North Carolina system as the first Executive Director of the UNC Teaching and Learning with Technology Collaborative. He has a passion for developing effective instructional content, and learning environments that promote rather than hinder student learning.