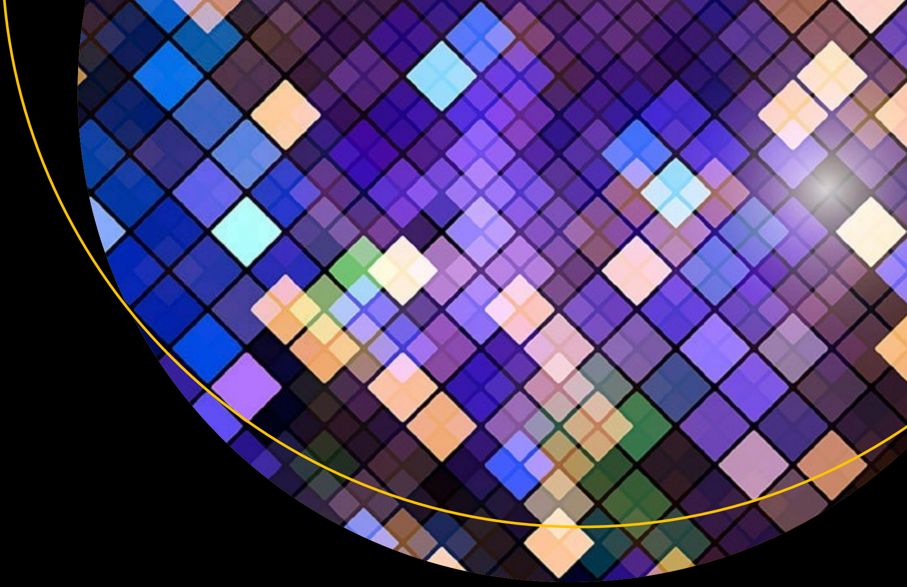

JavaScript Essentials for SAP ABAP Developers

A Guide to Mobile and Desktop Application Development

Rehan Zaidi

Apress®

JavaScript Essentials for SAP ABAP Developers

A Guide to Mobile and Desktop Application Development

Rehan Zaidi

Apress®

JavaScript Essentials for SAP ABAP Developers: A Guide to Mobile and Desktop Application Development

Rehan Zaidi
Karachi, Pakistan

ISBN-13 (pbk): 978-1-4842-2219-5
ISBN-13 (electronic): 978-1-4842-2220-1
DOI 10.1007/978-1-4842-2220-1

Library of Congress Control Number: 2017946310

Copyright © 2017 by Rehan Zaidi

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Cover image designed by Freepik

Managing Director: Welmoed Spahr
Editorial Director: Todd Green
Acquisitions Editor: Susan McDermott
Development Editor: Laura Berendson
Technical Reviewer: Diego Dora
Coordinating Editor: Rita Fernando
Copy Editor: Bill McManus

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail rights@apress.com, or visit http://www.apress.com/rights-permissions.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484222195. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

I dedicate this book to my Mother

Contents at a Glance

Contents

About the Author

Rehan Zaidi is a consultant for several international SAP clients (both onsite and remotely) on a wide range of SAP technical and functional requirements, and also provides writing and documentation services for their SAP- and ABAP-related products. He started working with SAP in 1999 and writing about his experiences in 2001. Rehan has written several articles for both *SAP Professional Journal* and *HR Expert*, and also has a number of popular SAP- and ABAP-related books to his credit.

Rehan is co-founder of IMZ Technologies, which provides SAP consulting to companies and helps clients (both onsite and remotely) with their SAP technical requirements (ABAP, Workflow, Quick development of Fiori apps, and S/4 HANA-related requirements). Rehan also creates documentation and training manuals for a number of companies based in the United States.

Rehan has clients located in a number of countries and continents, including the Middle East (GCC region), North America, and Europe. He also is currently working on a new ERP programmer magazine.

Rehan may be contacted via email at `rehan@imztechno.com` and `erpdomain@gmail.com`.

About the Technical Reviewer

Diego Dora is the Managing Director of sovanta AG's labs in Buenos Aires, Argentina. He has over 15 years' experience in the IT industry wearing different hats, including Developer, Software Architect, Development Manager, and Project Manager. Throughout his career, Diego has focused on developing enterprise business applications for international companies. He has extensive experience in JavaScript, SAPUI5, SAP Systems, SAP ABAP, SAP Cloud Platform, SAP HANA, and SAP Fiori, among other SAP technologies. Currently he is researching the subject of machine learning in the modern enterprise.

Acknowledgments

I am very thankful to both of my parents, in particular my mother (my main source of motivation and strength in life), and all my friends and well-wishers for their best wishes and prayers.

I am grateful to Susan McDermott for providing me the opportunity to write another SAP-related book that will be published from the United States. Many thanks to Rita Fernando for all the help and guidance during the writing and editing process.

I am very grateful to Diego Dora for his technical edits. Despite his personal problems and difficulties, he helped us make this book ready for publishing.

And I am very indebted to all the people of Apress who have worked on the development of this book.

Introduction

Mobile and desktop application development for SAP Fiori is a very important and popular topic for SAP developers. It requires SAP development tools based in JavaScript (JS). A major problem faced by many members of the SAP ABAP community is that they have little to no knowledge of JS. This book addresses that problem. Written as a quick guide for SAP ABAP developers to easily master JavaScript, this book will equip you with the necessary skills to develop mobile and desktop applications.

The primary emphasis of this book is on the parts of the JS language that are useful from the perspective of an ABAP developer. The book starts with a brief introduction to HTML, the basics of JS, and how to create and run a simple JS program. It then dives into the details of the language, showing you how to make simple programs. Next, it covers in detail loops, mathematical operations, and string and regular expression in JS. The book then gives you a taste of functions, followed by objects and object-oriented programming in JavaScript. Code examples and screenshots are provided throughout the book to help you fully understand JS. Finally, this book includes a chapter on miscellaneous topics, including JS best practices and recommendations.

This book is intended for SAP professionals, ABAP users, and university students. A brief overview of the chapters follows:

> Chapter 1: Introduction to JavaScript for ABAP. This chapter is an introduction to JavaScript for ABAP developers. It starts with the prerequisite knowledge for ABAP developers learning JavaScript. It next covers the basics of JavaScript, followed by the typical differences between the ABAP and JavaScript languages. Finally, it presents a very simple running JS program and shows you the necessary steps to create it yourself.
>
> Chapter 2: Getting Your Feet Wet with JavaScript Language. This chapter first gives you an overview of the `Window` object. Next, it describes in detail the data types that JavaScript provides. Then, demo programs will show you how to display data to the user in a dialog box, debug your web page in a web browser console, and receive input from the user via a dialog box. The final section provides a list of reserved words in the JS language. This chapter provides the foundation of the JS language.
>
> Chapter 3: Operators in JavaScript. As with all languages, the operators are very important for JS programmers and developers. This chapter starts with coverage of the arithmetic operators. Next, it introduces the comparison operators and logical operators that are used for formulating conditional checks in JS programs. Finally, the bitwise, assignment, and string operators are discussed.
>
> Chapter 4: Control Structures in JavaScript. Once you have a good grasp of the operators in JS, you will be ready to see the control structures provided by JavaScript in action. This chapter first looks at compound statements and coding examples. It then discusses the `if...else` statements and `switch` statements that are used for formulating conditions in programs. The chapter next covers in detail the loops applicable in JavaScript, such as the `for` loop, `for...in` loop, `while` loop, and `do...while` loop. Plenty of demo examples are provided to enable you to strengthen your understanding of the control structures.

Chapter 5: Regular Expressions. JavaScript provides the option of solving problems using regular expressions, which help fulfill requirements in a small number of code lines. This chapter discusses the RegExp object needed for working with regular expressions in JavaScript. It also covers the method of searching for and replacing patterns within a text stream. Finally, it provides actual problems and coding in JS using regular expressions.

Chapter 6: Functions in JavaScript. Functions play an important role in JS. This chapter begins with an overview of functions in JavaScript and the advantages they provide. You will see the syntax required to create functions, along with a simple program containing a function definition. Separate sections will show you how to create a program that calls a function and how to create a function that calls another function. In addition, you will see typical coding examples and requirements met via functional programming.

Chapter 7: More on Functions. Because functions are a major topic, two chapters are devoted to their coverage. Building on the coverage of the basics in Chapter 6, this chapter focuses on the advanced topics within the functions landscape. First, you will see how to use functions as variables. Then, you will be introduced to the concept of self-executing functions. Next, you will learn about nested functions—functions within functions. The chapter closes, appropriately, with a discussion of the very useful "closure" concept.

Chapter 8: Objects in JavaScript and Programming. As with ABAP, JavaScript allows you to create objects. However, the concepts pertaining to object creation in the two languages are not the same. This chapter is dedicated to object creation, instantiation, and inheritance. It starts with a general overview of object-oriented programming before drilling down to objects in JS and their syntax. It introduces the `this` operator in detail, then covers the various ways of creating objects, followed by the instantiation steps using the `new` keyword. Once you have a basic understanding of objects, the chapter moves on to inheritance as implemented in JS using prototypes. The chapter contains real-life examples of objects and subobjects throughout, along with ample code listings demonstrating how to implement them in JavaScript.

Chapter 9: Other Useful Objects in JavaScript. By this point in the book, you will be familiar with quite a few built-in objects in JavaScript. This chapter introduces a number of other useful objects, such as `Array`, `Boolean`, `Date`, and `String`, and the methods they provide. You will see several programming examples that implement these methods.

Chapter 10: Working with JSON. This chapter covers the important topic of JavaScript Object Notation. To begin, you will be introduced to JSON and see some typical examples. You will then see how the JSON arrays and objects work in JS programs. The chapter then covers two important functions, `stringify` and `parse`, used for programming with JS. As in all chapters, ample programming examples will be provided.

Chapter 11: Miscellaneous Topics. Chapter 11 contains several JS topics that are very useful but do not fit well in any of the other chapters. It starts with an overview of strict mode and how it affects the syntax check. Next, it covers error handling in JavaScript. A separate section covers the typical errors that you should avoid while working with JS. The chapter then explains debugging and troubleshooting JS programs. Last but not least, best practices for JS programming and tips for performance improvement are provided.

CHAPTER 1

Introduction to JavaScript

This chapter serves an introduction to JavaScript for ABAP developers. We will start with the prerequisite knowledge for learning JavaScript. Then, we will cover the basics of JavaScript, followed by some of the differences between ABAP and JavaScript. Next, we will look at a very simple program and the necessary steps for you to create it yourself.

Brief JavaScript Background

JavaScript is a high-level, dynamic, untyped language standardized in the ECMAScript language specification. Together with Hypertext Markup Language (HTML) and Cascading Style Sheets (CSS), JavaScript is another important technology that originally powered the Web Applications Architecture. JavaScript (JS) is supported by all modern web browsers, and is used by the majority of websites and web applications today. In the beginning, typical examples for the use of JavaScript included creating interactive effects on web pages and conducting form data validations on websites. Since then, a number of libraries have been written in JS, such as jQuery.

Note There is a common misconception that JavaScript and Java are similar. They are unrelated and have different semantics. The syntax of JavaScript is derived from the C programming language.

Also, the JS language is used in server-side programming as a runtime engine for several different browser engines such as V8 (Google Chrome) and SpiderMonkey (Firefox). One of the most widely known and used JS runtime environments is called NodeJS.

JavaScript has some syntactical similarities with the C language such as the `switch` and `if` statements and the `while` and `do while` loops. In JavaScript, types are linked with values, rather than with variables. For example, a variable named `myvar` could be assigned a string first and subsequently rebound to a number.

JavaScript is a case-sensitive language, meaning any language keywords, variable names, and function names must be written with consistent capitalization. The keyword `case`, for example, must be written "case" and not "Case" or "CASE."

JavaScript is a *multiparadigm* language that allows imperative programming, object-oriented programming, and functional programming. It allows you to work with text, numbers, dates, and arrays. Regular expression processing capability is also provided in JavaScript. However, JS does not offer any networking or graphics features. JavaScript supports prototypes (in contrast to many other object-oriented languages, such as ABAP) that use classes for inheritance. Many class-based features may be programmed using prototypes in JavaScript.

© Rehan Zaidi 2017

R. Zaidi, *JavaScript Essentials for SAP ABAP Developers*, DOI 10.1007/978-1-4842-2220-1_1

Inside a JavaScript Program

This section introduces the various different code elements of a JavaScript program in detail.

A JavaScript program is composed of a number of statements. Each statement ends with a semicolon (;). JavaScript differentiates between an expression and a statement. A statement may be composed of literals, variables, and expressions. A given JS statement might span multiple lines. Also, it is possible for more than one statement to be written in a single line.

Comments form an important part of any code. They are disregarded by JavaScript, but they are an essential tool for programmers to document the purpose of their code for later review. There are two types of comments you can add to a JavaScript program. First, you can add a single-line comment as shown here:

```
var num = 1; // from here comments start
```

As you can see, single-line comments begin with a double slash (//). All subsequent code/text written on that line is treated as comment text. No ending punctuation is necessary to mark the end of a single-line comment; the end of the line serves that purpose.

JavaScript also allows you to specify multiline comments. These may be enclosed between /* and */, and may comprise one or more lines. Within multiline comments, nesting is not possible. Attempting to nest multiline comments will produce a syntax error. Listing 1-1 shows an example that is not acceptable in JS.

Listing 1-1. Incorrectly Nested Comments

```
/*  this is comment 1
/*  this is comment 2  */
*/
```

As you can see, this example attempts to nest comment 2 within comment 1, which is not allowed and will give a syntax error.

Within a JavaScript program, statements are executed line by line (i.e., sequentially). A set of statements may be grouped together to form a block. The start and end of a block are denoted by curly brackets ({ }). These blocks may include code within a function (functions are similar to subroutines in ABAP) or code within `if` statements or `switch` statements.

A JS program may include a number of literals. You use literals in JS to denote fixed values (in contrast to variables) to be used within your programs. Some examples of these literals are integers and string literals.

In JavaScript, string literals are enclosed within a set of double quotes ("") or a set of single quotes (''), whereas ABAP strings are enclosed within single quotes only. Some other types of literals are array literals, Boolean literals, and object literals.

Creating a Simple JS Program

Now that you have some basic knowledge of JavaScript, this section shows you how to make a simple program, and then explains the various parts of it.

You will create a small program that displays a message saying "My first JS Program" in a pop-up dialog box. There is a built-in function `alert` in JS that lets you display a message to the user. This function may take as input as either a number or string in the form of a literal or variable.