LEARNING applescript

#applescript

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: applescript

It is an unofficial and free applescript ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official applescript.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with applescript

Remarks

AppleScript is a language that predates the Mac as we know it today, originating in 1993. While other scripting languages have more fully featured libraries and more robust syntax, the power of AppleScript is for automating Mac applications.

Any Mac application can include a scripting dictionary, and many that ship with macOS do. A scripting dictionary lets scripts know what items the app contains, and what actions can be performed.

A script can then use that information to automate repetitive tasks, and to form a "glue" between different apps, forming fully customizable workflows.

Versions

Version	OS Release	Release Date
1.6	10.0	2001-03-24
1.7	10.1	2001-09-25
1.9	10.2	2002-08-23
1.9.2	10.3	2003-10-24
1.10	10.4	2005-04-29
2.0	10.5	2007-10-26
2.1	10.6	2009-08-28
2.2	10.7	2011-07-20
2.2.4	10.8	2012-07-25
2.3	10.9	2013-10-22
2.4	10.10	2014-10-16
2.5	10.11	2015-06-08

Examples

Your first AppleScript

1. Open Script Editor.

2.12.4

With Mac OS X Leopard and earlier, and OS X Yosemite and later, Script Editor is located at

```
/Applications/Utilities/Script Editor.app
```

2.12.4

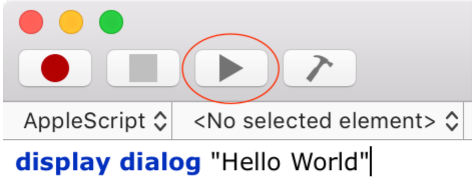
Between Mac OS X Snow Leopard and OS X Mavericks inclusive, Script Editor is AppleScript Editor.

```
/Applications/Utilities/AppleScript Editor.app
```

2. Enter the following line of code:

   ```
   display dialog "Hello World"
   ```

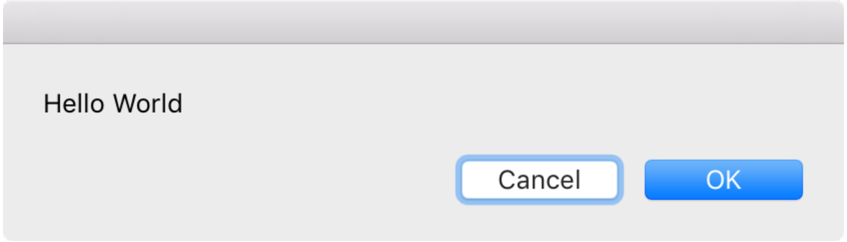
3. Click the run button.

   ```
   display dialog "Hello World"
   ```

   Two things happen: Script Editor compiles your script (if there are any errors, it'll let you know what's wrong and where), and then runs it.

4. You will now be shown a dialog saying "Hello World".

   
   Hello World

You've completed your first script!

What the code does:

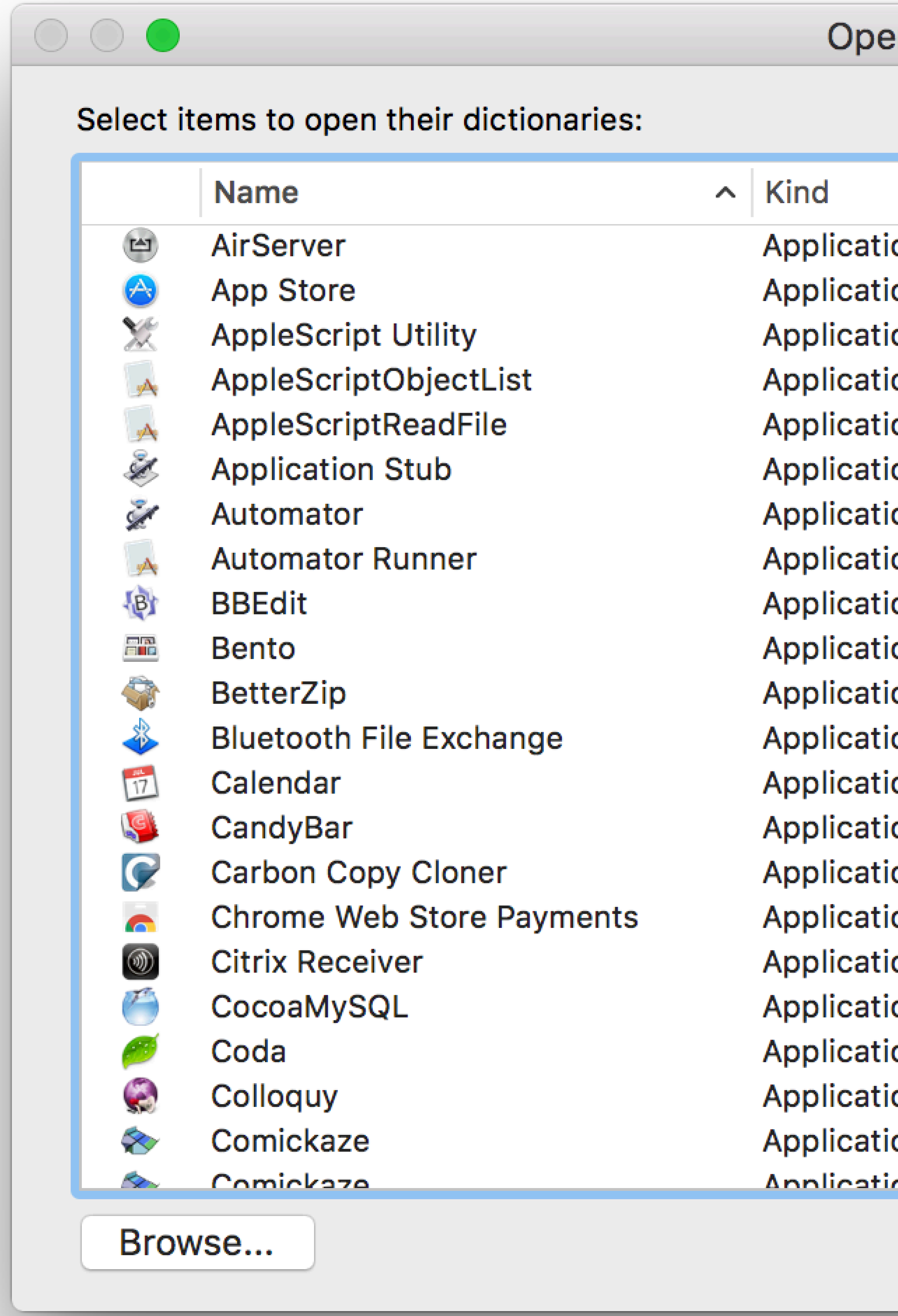
`display dialog` is one command, though it uses two words. This is common in AppleScript, unlike other languages which commonly require joining words together with no spaces. The `display dialog` command tells AppleScript to display a pop-up dialog.

This last part in "double quotes" tells the script what text to display.

Exploring scripting dictionaries

The power of AppleScript lies in being able to automate many Mac applications. To find out what you can automate, you need to read an app's scripting dictionary.

To do so, launch Script Editor, and select File > Open Dictionary…



Select items to open their dictionaries:

Name	Kind
AirServer	Applicatio
App Store	Applicatio
AppleScript Utility	Applicatio
AppleScriptObjectList	Applicatio
AppleScriptReadFile	Applicatio
Application Stub	Applicatio
Automator	Applicatio
Automator Runner	Applicatio
BBEdit	Applicatio
Bento	Applicatio
BetterZip	Applicatio
Bluetooth File Exchange	Applicatio
Calendar	Applicatio
CandyBar	Applicatio
Carbon Copy Cloner	Applicatio
Chrome Web Store Payments	Applicatio
Citrix Receiver	Applicatio
CocoaMySQL	Applicatio
Coda	Applicatio
Colloquy	Applicatio
Comickaze	Applicatio
Comickaze	Applicatio

https://riptutorial.com/applescript/topic/5271/getting-started-with-applescript

Chapter 2: AppleScript Browser Interactions

Introduction

AppleScript is able to retrieve the URL of the current tab of a browser.

Examples

Returning tab URLs

Safari

To return the URL of the current tab in Safari, use `URL of current tab`:

```
tell application "Safari"
    return URL of current tab of window 1
end tell
```

Google Chrome

To return the URL of the current tab in Google Chrome, use `URL of active tab`:

```
tell application "Google Chrome"
    return URL of active tab of window 1
end tell
```

Read AppleScript Browser Interactions online: https://riptutorial.com/applescript/topic/9030/applescript-browser-interactions

Chapter 3: Applescript from the Terminal command line

Introduction

Applescript is a powerful scripting language that can be used directly from the Terminal command line to accomplish a multitude of tasks.

All examples listed in this section are meant to be used from the Terminal application.

Examples

Get the current URL in Safari or Google Chrome

From the **Terminal** command line

Get the *current* URL from **Safari**

```
osascript -e 'tell app "safari" to get the url of the current tab of window 1'
```

Get the *active* URL in **Google Chrome**

```
osascript -e 'tell app "google chrome" to get the url of the active tab of window 1'
```

Get the Title of the current page in Safari or Google Chrome

Get the *name* of the web page in **Safari**

```
osascript -e 'tell app "safari" to get the name of the current tab of window 1'
```

Get the *title* of the web page in **Google Chrome**

```
osascript -e 'tell app "google chrome" to get the title of the active tab of window 1'
```

Use an Applescript as a shell function

You're not limited to single line Applescript code. Here we take the previous two examples and combine them into a single function.

```
#!/bin/bash

pageinfo() {
  osascript -e \
  'tell app "safari"
```

```
    tell the current tab of window 1
      return {url & "\n" & name}
    end tell
  end tell'
}
```

Read Applescript from the Terminal command line online: https://riptutorial.com/applescript/topic/10886/applescript-from-the-terminal-command-line

Chapter 4: AppleScript User Interaction

Syntax

- **display dialog** *text* [default answer *text*] [hidden answer *boolean*] [buttons *list of text*] [default button *text/integer*] [cancel button *text/integer*] [with title *text*] [with icon *text/integer/stop/note/caution/file*] [giving up after *integer*]
- **display alert** *text* [message *text*] [as *critical/informational/warning*] [buttons *list of text*] [default button *text/integer*] [cancel button *text/integer*] [giving up after *integer*]
- **display notification** *text* [with title *text*] [subtitle *text*] [sound name *text*]
- **choose file** [with prompt *text*] [of type *text*] [default location *alias*] [invisibles *boolean*] [multiple selections allowed *boolean*] [showing package contents *boolean*]
- **choose folder** [with prompt *text*] [default location *alias*] [invisibles *boolean*] [multiple selections allowed *boolean*] [showing package contents *boolean*]
- **choose from list** [with title *text*] [with prompt *text*] [default items *list of text/number*] [OK button name *text*] [cancel button name *text*] [multiple selections allowed *boolean*] [empty selection allowed *boolean]*
- **choose URL** [showing *list of Web/FTP/Telnet/File/News/Directory/Media/Remote*] [editable URL *boolean*]
- **choose color** [default color *RGB color*]

Remarks

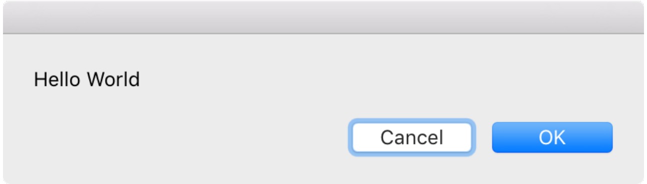
AppleScript User Interaction is part of Standard Additions. You can find the full documentation in the dictionary StandardAdditions.sdef through Script Editor's Open Dictionary.

Examples

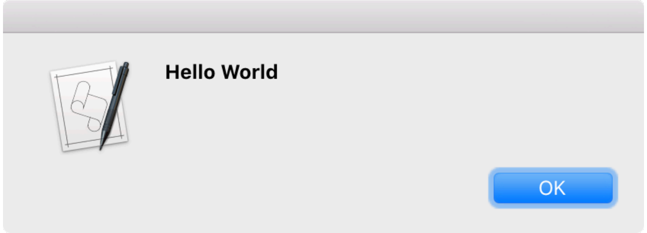
Display a dialog or alert

AppleScript can display dialogs and alerts to the user. Dialogs are for optionally requesting user input.

```
display dialog "Hello World"
display alert "Hello World"
```



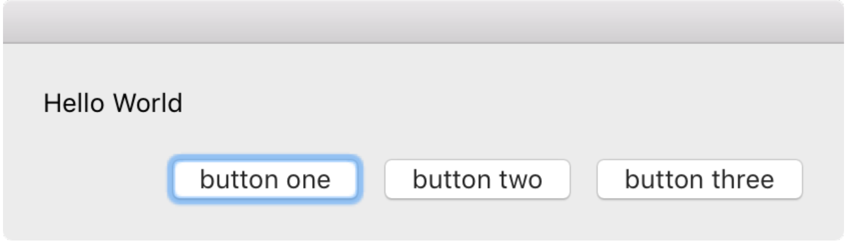
Hello World



Hello World

You can customise the buttons of either using `buttons` and passing a list of text.

```
display dialog "Hello World" buttons {"button one", "button two", "button three"}
```



Hello World

Upon clicking a button, the button clicked is returned:

```
{button returned:"button one"}
```

Read AppleScript User Interaction online: https://riptutorial.com/applescript/topic/6546/applescript-user-interaction

Chapter 5: AppleScript Variable Types

Examples

Numbers

```
set number1 to 8
```

Read AppleScript Variable Types online: https://riptutorial.com/applescript/topic/9975/applescript-variable-types

Chapter 6: Errors

Syntax

- **error** [*text*] [number *integer*]

Parameters

Parameter	Details
unnamed	A textual description of the error.
number	The error's number, an integer (usually negative).

Examples

Catching errors

Error handling in AppleScript uses **`try on error`**. The code which may throw an error goes in the `try` block and any error handling code is in the `on error` block. The `on error` block is closed using `end try`.

`foo` is not defined, so throws an error. When an error occurs, the dialog is displayed.

```
try
    foo
on error
    display dialog "An error occurred"
end try
```

It is possible to obtain the error message and error number using `on error` *`errormsg`* `number` *`errorno`* where *errormsg* and *errorno* are variable names for the error message and error number.

```
try
    foo
on error errormsg number errorno
    display dialog errormsg & errorno
end try
```

> The variable foo is not defined.-2753

Throwing errors

You can throw your own errors using `error`. Execution stops at uncaught errors. By default, the error message is "An error has occurred." with error number -2700.

```
error
```

```
error "An error has occurred." number -2700 from «script» to item
```

You can pass a message with the error which overrides the default message.

```
error "testing errors"
```

```
error "testing errors" number -2700 from «script» to item
```

Error numbers can also be passed using the parameter `number`:

```
error "testing errors" number -1
```

```
error "testing errors" number -1 from «script» to item
```

If you include an error number but no error message, an error message is written for you.

```
error number -1
```

```
error "An error of type -1 has occurred." number -1 from «script» to item
```

Some error numbers are reserved for certain types of error.

```
error number -42
```

```
error "Too many files open" number -42 from «script» to item
```

Read Errors online: https://riptutorial.com/applescript/topic/5404/errors

Chapter 7: Making Applescript If and Else Statements

Introduction

This topic is about if and else statements.

Examples

if variable = 2

```
set var to 2

if var = 2 then
    say "Var equals 2"
end if
```

if var1 = 4, else statements

```
set var1 to 5
//set the number to anything

if var1 = 5 then
    say "Var one equals 5"
else
 say "Var one does not equal 5"
end if
```

Dialogue returned text

```
display dialog "Password" default answer ""
set w to text returned of the result
if w = "Password" then
    display notification "Correct"
end if
```

Read Making Applescript If and Else Statements online: https://riptutorial.com/applescript/topic/9977/making-applescript-if-and-else-statements

Credits

S. No	Chapters	Contributors
1	Getting started with applescript	Community, Dov, grgarside, Pro Blaster
2	AppleScript Browser Interactions	grgarside, IT Master
3	Applescript from the Terminal command line	I0_ol
4	AppleScript User Interaction	grgarside
5	AppleScript Variable Types	SuperTech
6	Errors	dj bazzie wazzie, grgarside
7	Making Applescript If and Else Statements	SuperTech