NAME
> **platypus** — create Mac OS X application wrapper around a script.

SYNOPSIS
> **platypus** [**-vh**] [**-FASDBR**] [**-P** *profile*] [**-a** *appName*] [**-c** *scriptPath*]
>   [**-o** *outputType*] [**-i** *icon*] [**-p** *interpreter*] [**-V** *version*]
>   [**-s** *signature*] [**-I** *identifier*] [**-f** *bundledFile*] [**-X** *suffixes*]
>   [**-T** *filetypes*] [**-N** *environment*] [**-G** *arguments*] *appName.app*

DESCRIPTION
> **platypus** is the command line counterpart of the Platypus Mac OS X application. It is a command line too that creates Mac OS X application bundles that in turn execute a script.

> **platypus** supports the following flags (executed in the order which they are passed):

> **-p** *[profile]*
>> Loads all settings from a profile file generated by the Platypus.app Mac OS X application. You will still need to specify a destination path for the application.

> **-a** *[appName]*
>> Specifies the name of the application to be created. This is can be different from the name of the .app bundle itself, and is displayed in the application's menus, about window and Info.plist property list.

> **-c** *[scriptPath]*
>> Path to the script that is to be wrapped into an application.

> **-o** *[outputType]*
>> Specifies the script application's output type. Four different kinds of (case sensitive) values are supported as arguments:

>> *'None'* The application has no graphical output. Script text output goes to STDOUT.

>> *'Progress Bar'* The application displays an indeterminate progress bar. Script output goes to STDOUT.

>> *'Text Window'* The application displays a window with a text field which receives all script output.

>> *'Web'* The application displays a window with a web view which renders output as HTML.

> **-i** *[icon]*
>> Specifies a file to use as icon for the application bundle. The icon file can be a regular Mac OS X .icns file, or an image file of any kind supported by the Mac OS X Cocoa APIs. As of writing, this includes JPEG, GIF, PNG, TIFF, PSD, BMP, PICT, TGA, SGI, PNTG PDF, EPS and PostScript (.ps). The image will be scaled to 128x128 pixel size, which is standard for Mac OS X icons.

> **-p** *[interpreter]*
>> Sets the interpreter for your script (e.g. /usr/bin/perl). If no interpreter is specified, the default is the default shell (/bin/sh).

> **-V** *[version]*
>> Sets the application bundle's version. This is reflected in the application bundle's Info.plist property list and About window.

> **-u** *[author]*
>> Sets the name of the application author (e.g. "Apple Computer" or "John Smith"). If not specified, platypus will default to the current user's full user name.

−s *[signature]*
Mostly for backwards compatibility and heritage reasons. In Mac OS 9 and earlier (and, indeed, in Mac OS X), each application was assigned a four character ASCII signature known as an OSType. It is possible to register application signatures in a database on the Apple Developer Website. Please note that Apple has reserved all lowercase-only combinations.

Chances are that you will have no need of this option. If it is left empty, the application will get the signature string '????', which is ignored by Mac OS X. This value is stored within the application bundle in the Info.plist application propertly list.

−f *[bundledFile]*
Specifies a file to be bundled with the script application. This file will be copied over the Resources folder of the application bundle, which is the same folder in which your script will eventually reside. You can specify any number of files to be bundled, but you must use the -f flag each time, since the flag only accepts one file path after the -f flag.

−I *[identifier]*
Sets the application's bundle identifier. An application identifier is a kind of reverse DNS name (e.g. com.apple.iTunes). If this option is left empty, **platypus** will default to an identifier of the format "org.username.appname" (e.g. org.sveinbjornt.Platypus).

−F If set, your script will receive the path to the application (e.g. "/Applications/MyPlatypusApp.app") as the first argument ($ARGV[1], $1 etc., depending on your scripting language of choice).

−A This flag makes the application request Administrator privileges via Apple's Security Framework (i.e. prompt for a password) and then execute the script with those privileges.

−S Secure bundled script. This encrypts the bundled script, to fend off prying eyes.

Please note that this will NOT make the script inaccessible to an experienced hacker, but just might serve to dissuade the less determined.

−D Makes your script application "droppable", i.e. capable of receiving drag'n dropped files as arguments to the script. The application bundle's property list is modified so that it can receive dropped files in the Dock and Finder. These files are then passed on to the script as arguments via @ARGV.

−B This option causes the application to run in the background. The application will not appear in the Dock or bring its windows to the front when launched. This is achieved by registering the application with the Finder as a user interface element (LSUIElement).

−R This option sets the application so that it remains running after the script has been executed. It can then later be quit by the user via regular means (i.e. Quit menu item).

−X *[suffixes]*
Only appropriate if you are using the -D option. This flag allows you to specify the file suffixes (e.g. .txt, .wav) your application can open. This should be a |-separated string (e.g. "txt|wav|jpg").

−T *[filetypes]*
Only appropriate if you are using the -D option. This flag allows you to specify the file type codes (e.g. '8BIM', 'TEXT') your application can open. This should be a |-separated string of 4-character ASCII strings (e.g. "8BIM|TEXT|R∗ch").

−G *[arguments]*
Arguments for the script interpreter. These should be specified as a |-separated string (e.g. '-w|-s|-l').

−N *[environemnt]*
Sets environmental variables for your script. These should be specified as a |-separated string of name=value pairs (e.g. 'EDITOR=pico|TERM=xterm-color').

−v   Prints the version of the platypus command line utility

−h   Prints help and usage string

The **platypus** utility exits 0 on success, and >0 if an error occurs.

FILES
```
/usr/local/bin/platypus                         program binary
/usr/local/share/platypus/ScriptExec            executable for script app
/usr/local/share/platypus/MainMenu.nib          Nib file for script app
/usr/local/share/platypus/PlatypusDefault.icns  Default icons
```

AUTHORS
This manual page was written by Sveinbjorn Thordarson ⟨sveinbjornt@simnet.is⟩.