

Riding Your Next Mac Admin Wave

Tim Sutton
Sauce Labs

MacSysAdmin crew

Tycho Sjögren

Patrik Jerneheim

Lothar Molin

Anne Sjögren

Tommy Suto

Nikolai Waldman

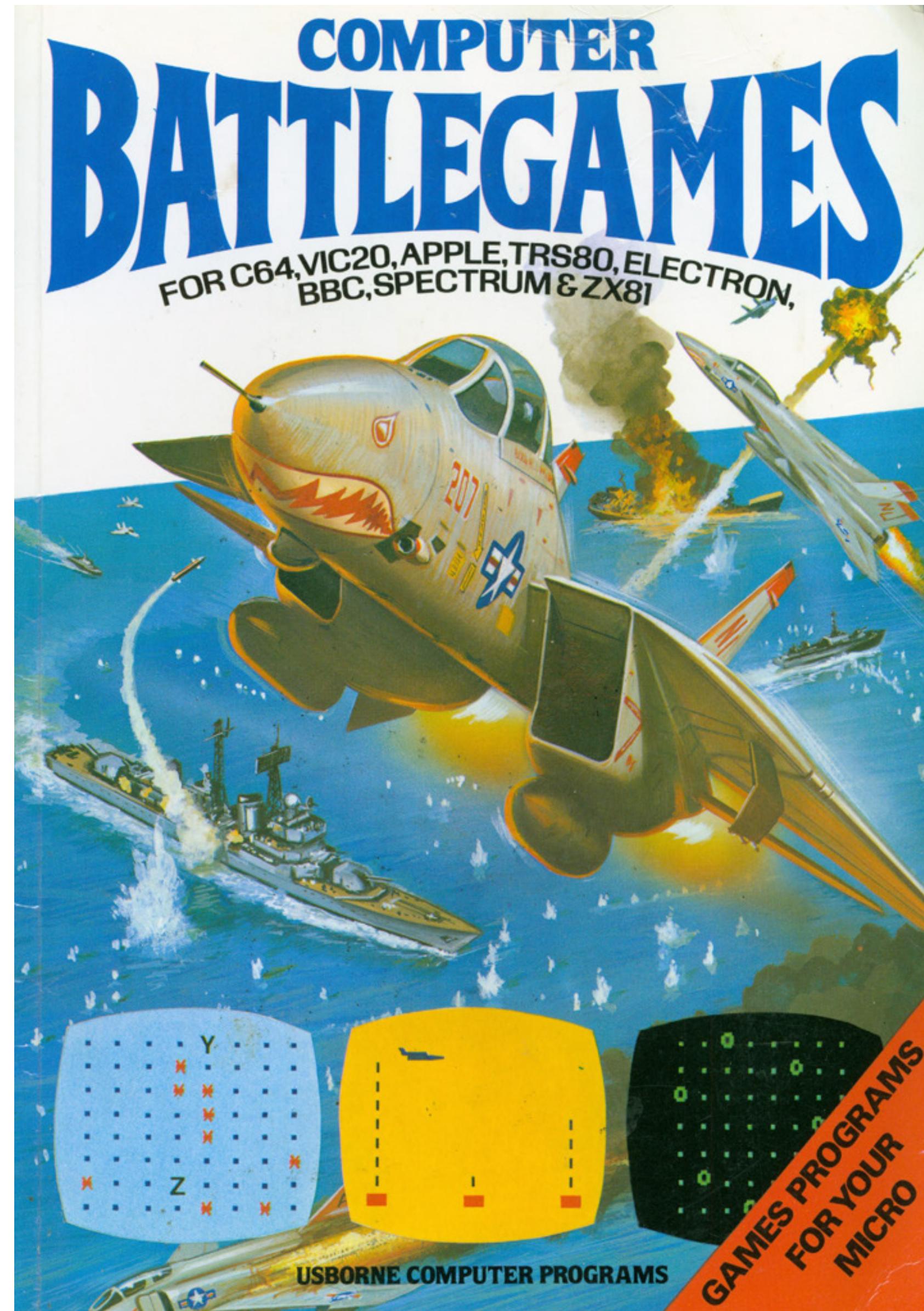
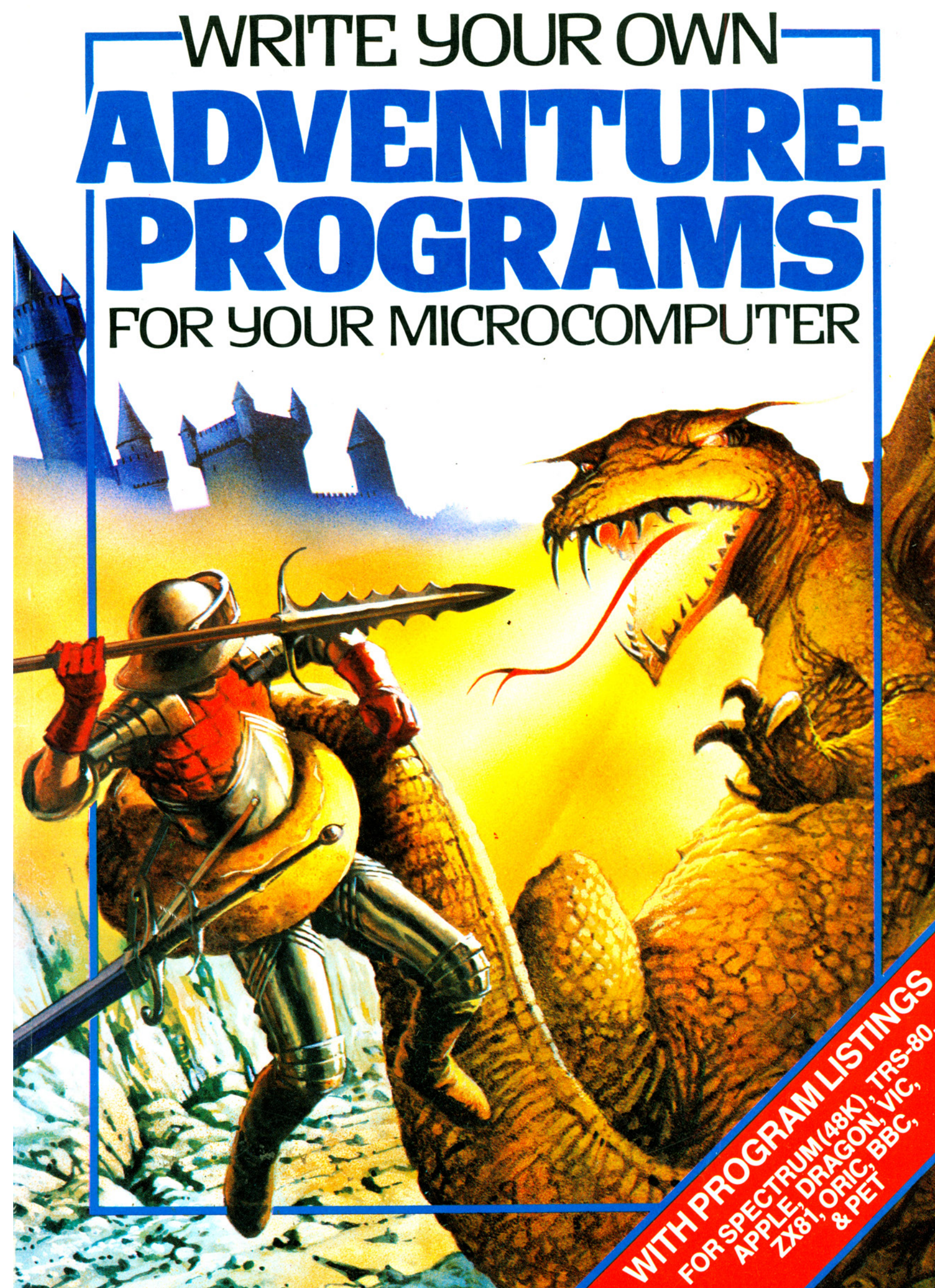


W. Andrew Robinson

Thomas Norrgård

Ben Goodstein

Usborne computer books (1982)



Usborne computer books (1982)



How the program works

```
▲●5 CLS
10 PRINT "EVIL ALIEN"
20 LET S=10
30 LET G=4
★▲●40 LET X=INT(RND*S)
★▲●50 LET Y=INT(RND*S)
★▲●60 LET D=INT(RND*S)

70 FOR I=1 TO G

80 PRINT "X POSITION (0 TO 9)?"
85 INPUT X1
90 PRINT "Y POSITION (0 TO 9)?"
100 INPUT Y1
110 PRINT "DISTANCE (0 TO 9)?"
120 INPUT D1

130 IF X=X1 AND Y=Y1 AND D=D1 THEN GOTO 300
140 PRINT "SHOT WAS ";
150 IF Y1>Y THEN PRINT "NORTH";
160 IF Y1<Y THEN PRINT "SOUTH";
170 IF X1>X THEN PRINT "EAST";
180 IF X1<X THEN PRINT "WEST";
190 PRINT
200 IF D1>D THEN PRINT "TOO FAR"
210 IF D1<D THEN PRINT "NOT FAR ENOUGH"

220 NEXT I

230 PRINT "YOUR TIME HAS RUN OUT!!"
240 STOP
300 PRINT "*BOOM* YOU GOT HIM!"
310 STOP
```

— Sets the size of the grid.

— Sets the number of goes.

— Elron's position is fixed by these 3 lines, which select 3 numbers between 0 and the size of the grid.

— Start of a loop which tells the computer to repeat the next 15 lines G times.

— This section asks you for your 3 numbers and stores them in X1, Y1 and D1.

— Checks if you were right and jumps to 300 if you were.

— Your guesses are compared with Elron's position and a report printed.

— End of loop. Returns for another go.

— Prints if you've used up all your goes.

— Prints if you guessed right.

How to make the game harder

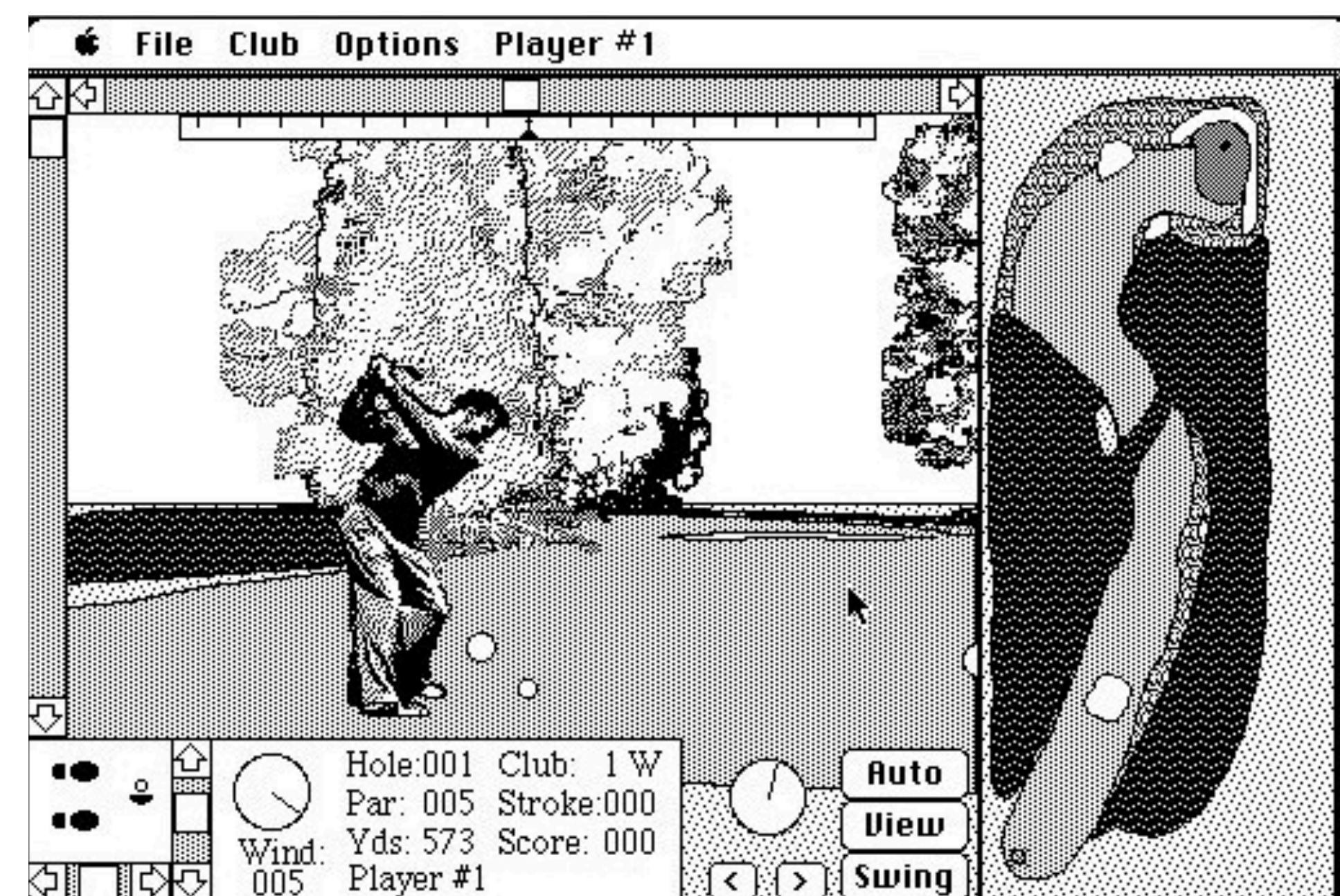
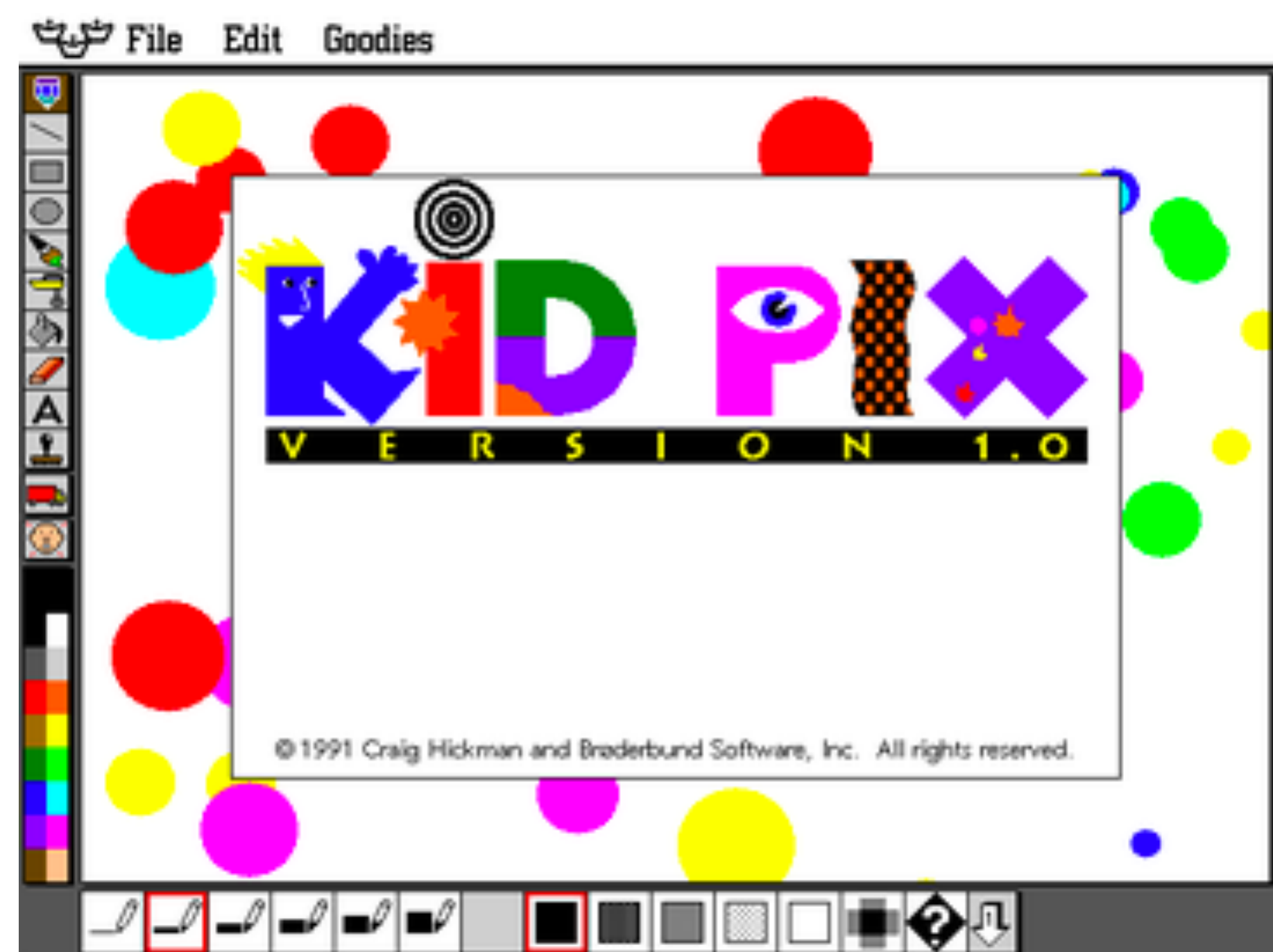
This program has been written so that you can easily change the difficulty by changing the size of the grid. To do this, put a different value for S in line 20.

If you increase the grid size you will need more space bombs to give you a fair chance of blasting Elron. Do this by changing the value of G in line 30.

Puzzle corner

Can you work out how to change the program so that the computer asks you for a difficulty number which it can put into S instead of S being fixed? (Tip: limit the value of S to between 6 and 30 and use INT(S/3) for the value of G in line 30.)

See page 2 for meaning of ●▲★



0 / 100

concurrent sessions

Watch **Commands** Logs Metadata

FILTER: Command ☐ Has Screenshot

▶ Play ⏮ ⏭ 2 of 3

| | | |
|--|---------------------|--------------------------|
| POST /session | 00:00:00 + 4.65s | <input type="checkbox"/> |
| POST url | 00:07:69 + 3.71s | <input type="checkbox"/> |
| DELETE /session/1f91f125d01046f9b93340473... | 00:29:87 + 0.00s | <input type="checkbox"/> |

00:00:07 00:00:29

Safari File Edit View History Bookmarks Develop Window Help

reliawire.com/banana-nutrition-facts/

RELIAWIRE

Neuroscience Neurology Oncology Medicine

10 Banana Nutrition Facts You Should Know

by James Anderson ⌚ July 15, 2008

I love Bananas; who doesn't? They taste great, are easy to take anywhere and eat and are healthy. But are up to speed on all the nutrition data? Check

11 SHARES

f

🐦

G+

📌

Follow Us

Top Posts This Week

Vagal Gut-brain Axis Stimulation Produces Reward Behaviors

Insights Into Autism From The Protein Modulator ASTN2

Ancient Retroviral Infection Linked To Addictive Behaviors

SynuClean-D Can Revert Neurodegeneration Caused By Parkinson's

Riding Your Next
^(Mac\w*Admin)*.*\$
Wave

Infrastructure

Integration

**What do you like to
build?**



pudquick / cms_detached_verify.py

Last active 4 months ago

Verifying a CMS detached signature in pyobjc on macOS

1 file 0 forks 0 comments 4 stars

```
1 import objc
2 from ctypes import create_string_buffer, c_void_p, cast
3 from Foundation import NSBundle
4
5 Security = NSBundle.bundleWithIdentifier_('com.apple.security')
6
7 # CMSDecoder.h
8 kCMSSignerUnsigned = 0
9 kCMSSignerValid = 1
10 kCMSSignerNeedsDetachedContent = 2
```

[View cms_detached_verify.py](#)



pudquick / xcode_vers.py

Last active 6 months ago

Programmatically load detailed version information about Xcode version via pyobjc on macOS

1 file 0 forks 0 comments 2 stars

```
1 # Warning - because of how this works, it loads classes into memory namespace.
2 # Attempting to load a second Xcode to inspect within the same python run will result in errors
3 # If you need to inspect multiple, for now, just spin the inspection up under a second process
4
5 from Foundation import NSBundle
6
7 def xcode_info(app_path):
8     # app_path = '/Applications/Xcode.app'
9     DVTFoundation = NSBundle.bundleWithPath_('%s/Contents/SharedFrameworks/DVTFoundation.framework' % app_path)
10    IDEKit = NSBundle.bundleWithPath_('%s/Contents/Frameworks/IDEKit.framework' % app_path)
```


Incremental

MacSysAdmin 2013



→ AutoDMG cloc .
43 text files.
42 unique files.
16 files ignored.

github.com/AlDanial/cloc v 1.78 T=1.03 s (27.2 files/s, 6092.5 lines/s)

| Language | files | blank | comment | code |
|--------------|-------|-------|---------|------|
| Python | 21 | 900 | 496 | 4040 |
| Bourne Shell | 3 | 39 | 33 | 290 |
| Objective C | 2 | 66 | 78 | 241 |
| C/C++ Header | 1 | 12 | 29 | 22 |
| Markdown | 1 | 10 | 0 | 12 |
| SUM: | 28 | 1027 | 636 | 4605 |

github.com/chilcote/vfuse

📖 README.md

vfuse

This script takes a never-booted DMG and converts it to a VMware Fusion VM.

The germ of this idea came about, as all good ideas, and germs, do: while drinking beer. Specifically, I was tossing back drinks and tossing around ideas with [Gilbert Wilson](#), and he mentioned that he uses the VMware CLI tools to convert DMGs to VMDKs based on a [blog post](#) he'd read. Intrigued, I asked Gil to email me the specifics. After seeing how potentially cool this was, I wrapped it up in this here terribly illegible, queasily unpythonic script.

My thanks to [Rich Trouton](#) for testing this script for me and providing feedback. He is a saint among humans.

Requirements

- VMware Fusion 10.x Professional
- OS X 10.9.5+
- A never-booted image created with [AutoDMG](#).
- (optional) [Packer](#) 1.1.1 (or above) for building a vagrant box.
- (optional) [qemu-img](#)

macOSLAPS - Randomized Local Admin Passwords - Joshua D. Miller



<https://www.youtube.com/watch?v=U3WgtP5i6JA>

NoMAD

April 2009 – Peter Bukowinski [announced](#) as project called “AD Password Monitor”

April 2011 – Rusty Myers reaches out to Peter about moving “AD Password Monitor” from a dashboard widget to an app. Soon afterwards, Peter [announced](#) the first release of ADPassMon, taking “AD Password Monitor” & turning it into a app.

April 2014 – I [announced](#) my fork of ADPassMon, this added a few additions to ADPassMon. Notably keychain stuff along the lines of Joel Rennich’s Keychain Minder. This was due to the addition of the [local items keychain](#) to macOS 10.9 & Keychain Minder not working with this new keychain. At the time Joel was working at Apple, so Keychain Minder so it became unmaintained.

December 2014 – Peter Bukowinski [announced](#) the release of KerbMinder.

September 2015 – My fork of ADPassMon was merged with Peter’s.

October 2015 – Francois Levaux-Tiffreau [works on a release of KerbMinder](#), that removes the requirement for the Mac to be bound to AD.

Around the same time. Peter, Francois & myself start talking on slack with Kyle Crawshaw about moving ADPassMon to Python, then creating a module that ADPassMon, KerbMinder & Kyle’s [ShareMounter](#) was use. In a project called: Gala.

November 2015 – Apples Professional Services team, announced Enterprise Connect. Within it’s Keynote deck, shown at webinars, ADPassMon is shown as an example of how to manage passwords when bound to AD.

December 2015 – Joel Rennich started a new role a TruSource after leaving Apple, & starts DM-ing me on [Slack](#) about possibly moving ADPassMon to Swift, I didn’t take the bait

Feb 2016 – Peter [took a role at Apple](#), & I become primary maintainer of ADPassMon.

April 2016 – I [joined dataJAR](#), who have no AD themselves. However some customers do use AD. This meant I was no longer [dogfooding](#) ADPassMon.

Around the same time, Kyle & Francois took new roles. Or the focus of their roles changed, so that KerbMinder & ShareMounter were not such a high focus.

August 2015 – Joel makes first commit to [NoMAD](#).

November 2016 – Tom Nook adds ShareMounter preferences to [NoMAD](#).

December 2016 – [NoMAD](#) release candidates start being released.

March 2017 – [NoMAD](#) 1.0.3 released.

<https://macmule.com/2017/04/01/adpassmon-is-dead-long-live-nomad/>

What tools?

- Management agents
- Configuration tools
- Onboarding / enrolment / assistants
- Access to resources
- Inventory collection
- Backend integrations

**Measure, know,
assess**

Bash, Python

Logging

Abstract the log function

```
format_log() {  
    msg="$1"  
    level="$2"  
    echo "$(date +%Y-%m-%dT%H:%M:%S%z) - ${level} - ${msg}"  
}
```

```
log_info() {  
    format_log "$1" "INFO"  
}
```

```
log_error() {  
    # log to stderr instead of stdout  
    >&2 format_log "$1" "ERROR"  
}
```

Abstract the log function

```
import logging
import os

logging.basicConfig(format="%(asctime)s - %(levelname)s - %(message)s",
                    level=logging.INFO)
log = logging.getLogger('myapp')

def set_up_dirs():
    log.info("Doing stuff")

    try:
        os.mkdir('/tmp/scratch')
    except OSError as err:
        log.error("Oh no! error message: %s", err)

set_up_dirs()
```


Abstract the log function

```
{ 'args': (OSError(17, 'File exists'),),  
  'created': 1538671174.334487,  
  'exc_info': None,  
  'exc_text': None,  
  'filename': 'logs.py',  
  'funcName': 'set_up_dirs',  
  'levelname': 'ERROR',  
  'levelno': 40,  
  'lineno': 17,  
  'message': "Oh no! error message: [Errno 17] File exists: '/tmp/scratch'",  
  'module': 'logs',  
  'msecs': 334.4869613647461,  
  'msg': 'Oh no! error message: %s',  
  'name': 'myapp',  
  'pathname': 'logs.py',  
  'process': 82529,  
  'processName': 'MainProcess',  
  'relativeCreated': 2.8460025787353516,  
  'thread': 140735508587392,  
  'threadName': 'MainThread' }
```

Log aggregation

User interface / Documentation

Resilience

Error handling

```
>>> import os
```

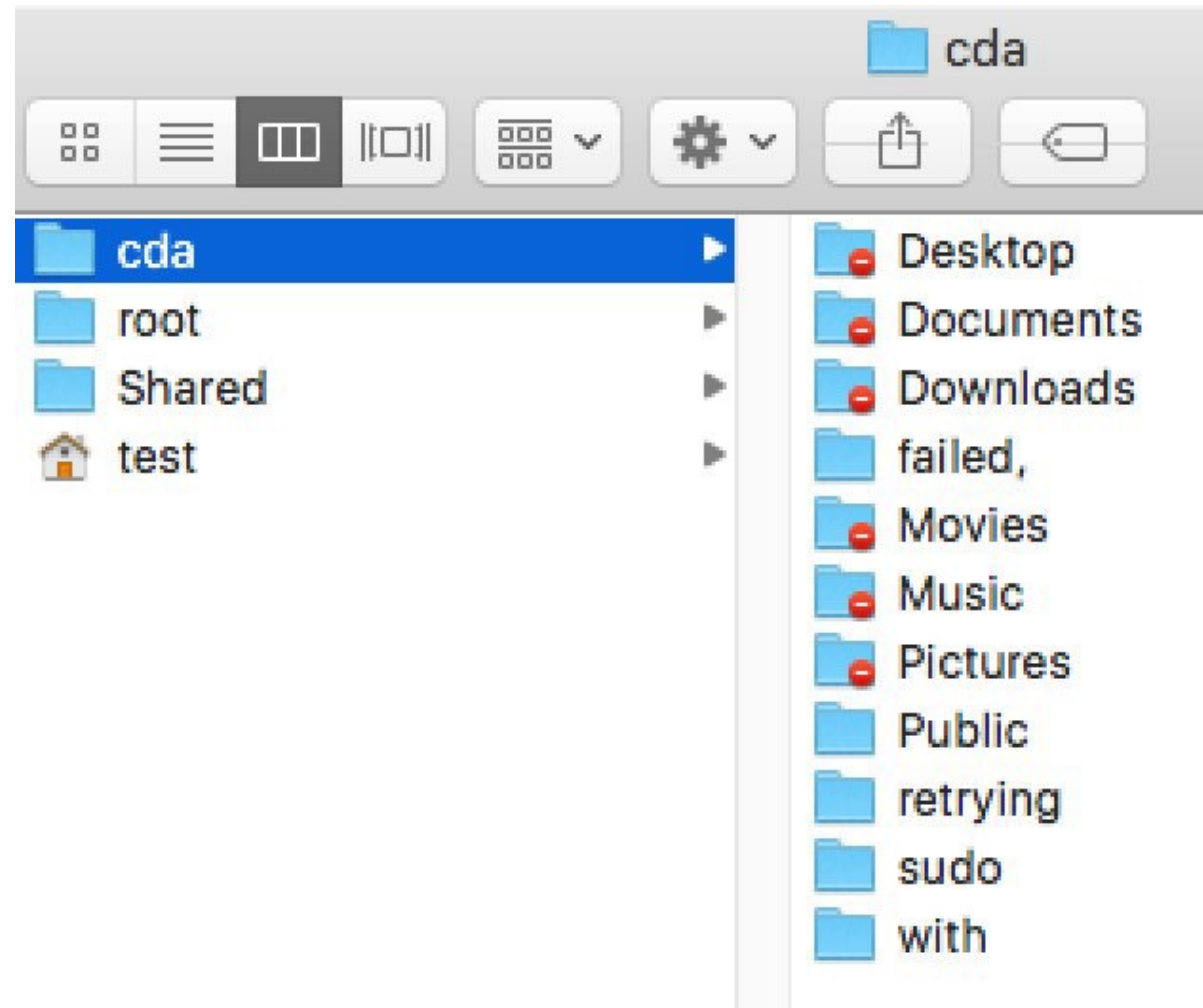
```
>>> os.mkdir('/tmp/scratch')
```

```
Traceback (most recent call last):
```

```
  File "<stdin>", line 1, in <module>
```

```
OSError: [Errno 17] File exists: '/tmp/scratch'
```

→ / mkdir /tmp/scratch
mkdir: scratch: File exists
→ /



```
>>> import os
```

```
>>> if not os.path.exists('/tmp/scratch'):  
...     os.mkdir('/tmp/scratch')
```

```
#!/bin/bash -eu
```

(or)

```
set -eu
```


Trap (bash)

```
#!/bin/bash -eu
```

```
cleanup() {  
    hdiutil detach -quiet -force "$MNT_ESD" || echo > /dev/null  
    hdiutil detach -quiet -force "$MNT_BASE_SYSTEM" || echo > /dev/null  
    rm -rf "$MNT_ESD" "$MNT_BASE_SYSTEM" "$BASE_SYSTEM_DMG_RW" "$SHADOW_FILE"  
}
```

```
trap cleanup EXIT
```

```
if [ ! -e /tmp/scratch ]; then  
    mkdir /tmp/scratch  
fi
```

```
echo 'doing more things..'
```

try/except/finally (python)

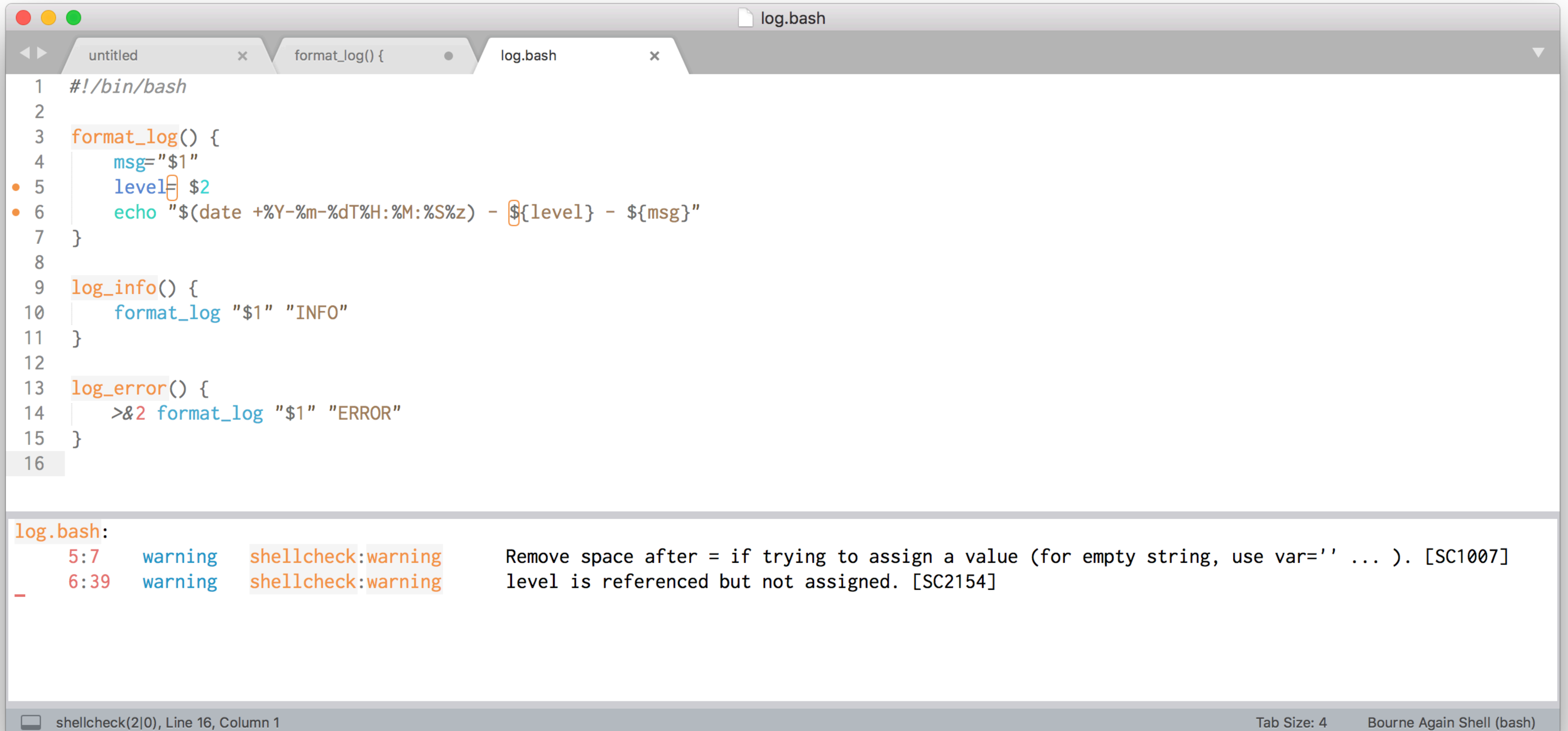
```
def divide(x, y):  
    try:  
        result = x / y  
    except ZeroDivisionError:  
        print "division by zero!"  
    else:  
        print "result is", result  
    finally:  
        print "executing finally clause"
```

In real world applications, the finally clause is useful for releasing external resources (such as files or network connections), regardless of whether the use of the resource was successful.

<https://docs.python.org/2/tutorial/errors.html#defining-clean-up-actions>

**Validate user input,
fail early**

Linters



The screenshot shows a code editor window titled "log.bash" with three tabs: "untitled", "format_log() {", and "log.bash". The script content is as follows:

```
1 #!/bin/bash
2
3 format_log() {
4     msg="$1"
5     level=$2
6     echo "$(date +%Y-%m-%dT%H:%M:%S%z) - ${level} - ${msg}"
7 }
8
9 log_info() {
10     format_log "$1" "INFO"
11 }
12
13 log_error() {
14     >&2 format_log "$1" "ERROR"
15 }
16
```

Below the script, the ShellCheck linting output is displayed:

```
log.bash:
  5:7  warning  shellcheck:warning  Remove space after = if trying to assign a value (for empty string, use var='' ... ). [SC1007]
  6:39 warning  shellcheck:warning  level is referenced but not assigned. [SC2154]
```

The status bar at the bottom indicates "shellcheck(2|0), Line 16, Column 1", "Tab Size: 4", and "Bourne Again Shell (bash)".

2017 Session Documentation

Copyright © 2017 MacSysAdmin AB and the respective presenter. All rights reserved.



OS State of the Union

Arek Dreyer, [Dreyer Network Consultants](#).



What do we actually care about?

Charles Edge, [Jamf](#) and [krypted.com](#)



The IT Workplace in the Agile Age

Andreas Westendörpf, [FileWave](#)



Cybersecurity in 2017! And what about 2018?

Sascha Uhl, [Code42](#)



Scripting Bash

Armin Briegel, [scriptingosx.com](#)



50/50

Duncan McCracken, [Mondada](#)



Open Source

Opportunities

sys admin/eng/cpe/etc 2018

- HTTP(S)
- REST
- Encryption, hardware and software
- Certificates
- Cloud-based, subscription licensing for software and services
- MFA
- Identity Providers
- Cloud web services.- AWS, Gcloud, DigitalOcean
- Swift, Golang, node.js, Ruby, Python
- Docker, Kubernetes
- Mobile / LTE
- Everything, everything connecting to Slack

Value

Go and do the thing

Sysadmins rule

Imposter Syndrome

...in which an individual doubts their accomplishments and has a persistent internalized fear of being exposed as a “fraud”

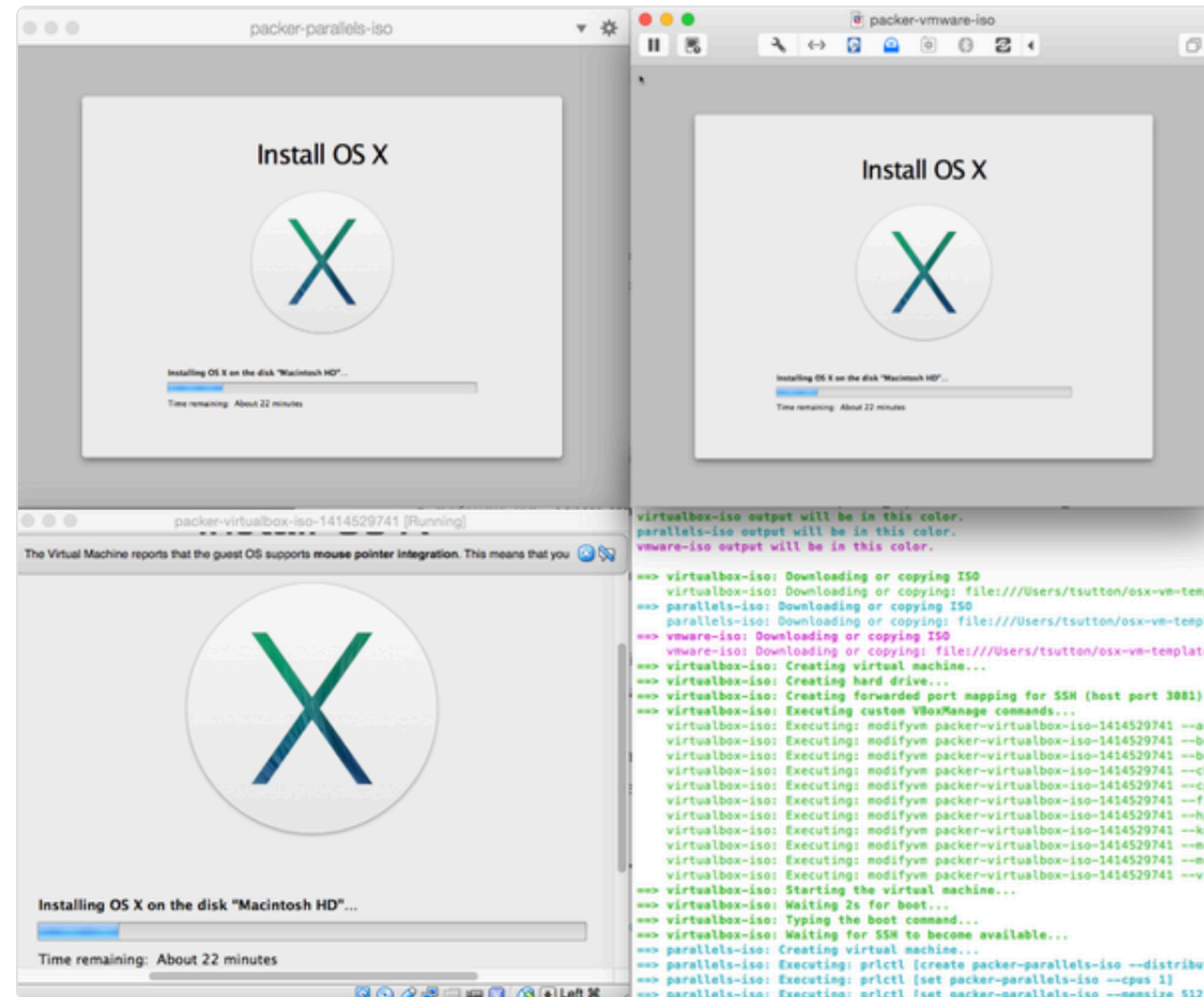
https://en.wikipedia.org/wiki/Impostor_syndrome

Tim Sutton

@tvsutton



Because we can: Packer simultaneously building OS X VMs on Fusion, Parallels and VirtualBox.



5:01 PM - 28 Oct 2014 from [Ville-Marie, Montréal](#)

2 Retweets 6 Likes



Tack!

<https://macops.ca/new-wave>

