abstraction distractions

NEAL FORD software architect / meme wrangler

ThoughtWorks®

nford@thoughtworks.com
3003 Summit Boulevard, Atlanta, GA 30319
www.nealford.com
www.thoughtworks.com
blog: memeagora.blogspot.com
twitter: neal4d

abstraction

distraction

Mary Long Chocolate Cake

- 1 3/4 cups all-purpose flour
- 2 cups white sugar
- 3/4 cup unsweetened cocoa powder
- 2 teaspoons baking soda
- 1 teaspoon baking powder
- 1 cup buttermilk
- 1 teaspoon vanilla extract
- …

?

plain text

Unicode ?   ASCII !

<html>

NATIONAL ARCHIVES AND RECORDS ADMINISTRATION
LITTERA SCRIPTA MANET
1985

The Clock of the Long Now Project

http://longnow.org/

HERE

imagine a
giant
clock...

Mary Long Chocolate Cake

abstraction

- 1 3/4 cups all-purpose flour
- 2 cups white sugar
- 3/4 cup unsweetened cocoa powder
- 2 teaspoons baking soda
- 1 teaspoon baking powder
- 1 cup buttermilk
- 1 teaspoon vanilla extract
- . . .

distraction

Lesson #1:

Don't mistake the abstraction for the real thing

And then we take the TPS reports and place a cover sheet on them. We used to use red covers, but the blue ones work better. The TPS report was tough because of the table structure in the data warehouse. We had to denormalize some of the columns and create some special indexes to get the performance fast enough because the CEO said he needs to TPS reports (with cover) by 9 AM...

rxstudios

night auditor responsibilities

- daily accounting
- late checkins
- help with guests
- don’t get drunk
- don’t sleep

now with electricity added!

Craptaculous Suites

Prev. Bal:	0.00
Room Chg:	89.00
Tax:	9.79
New Balance:	98.79

∑ keys

==

Δ prev/new balances

Craptaculous Suites

Prev. Bal:	0.00
Room Chg:	89.00
Tax:	9.79
New Balance:	98.79
Prev. Bal:	98.79
Gift Shop:	2.55
New Balance:	101.34

Craptaculous Suites

Prev. Bal:	0.00
Room Chg:	89.00
Tax:	9.79
New Balance:	98.79
Prev. Bal:	**89.79**
Gift Shop:	2.55
New Balance:	92.34

Craptaculous Suites

Prev. Bal: 0.00

Room Chg: 89.00

Tax: 9.79

New Balance: 98.79

Prev. Bal: 0.00

Gift Shop: 2.55

Prev. Bal: 98.79

Gift Shop: 2.55

Gift Shop: 2.55

New Balance: 98.79

New Balance: 101.34

Prev. Bal: 0

Room Chg: 89

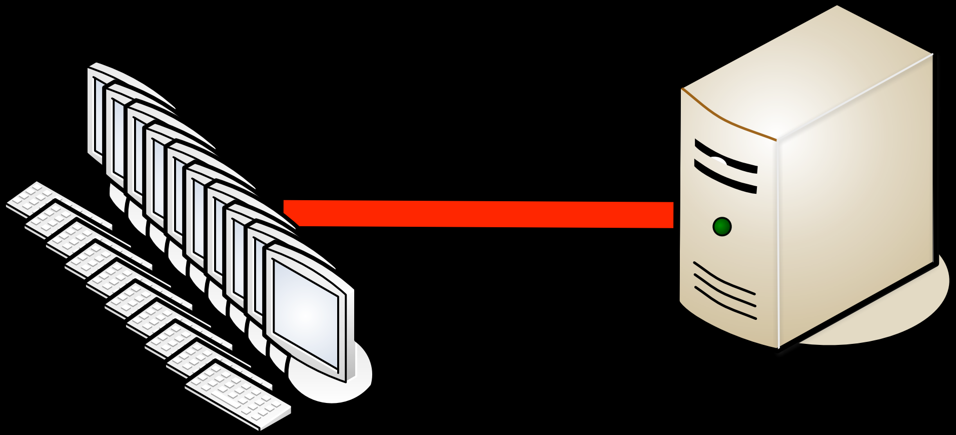
Tax: 9

New Balance: 98

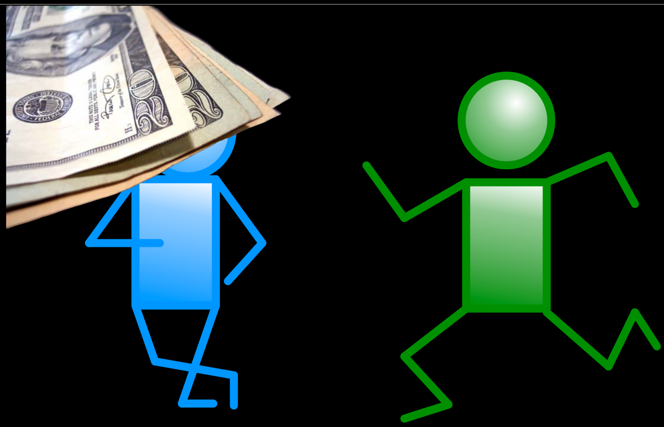
IBM

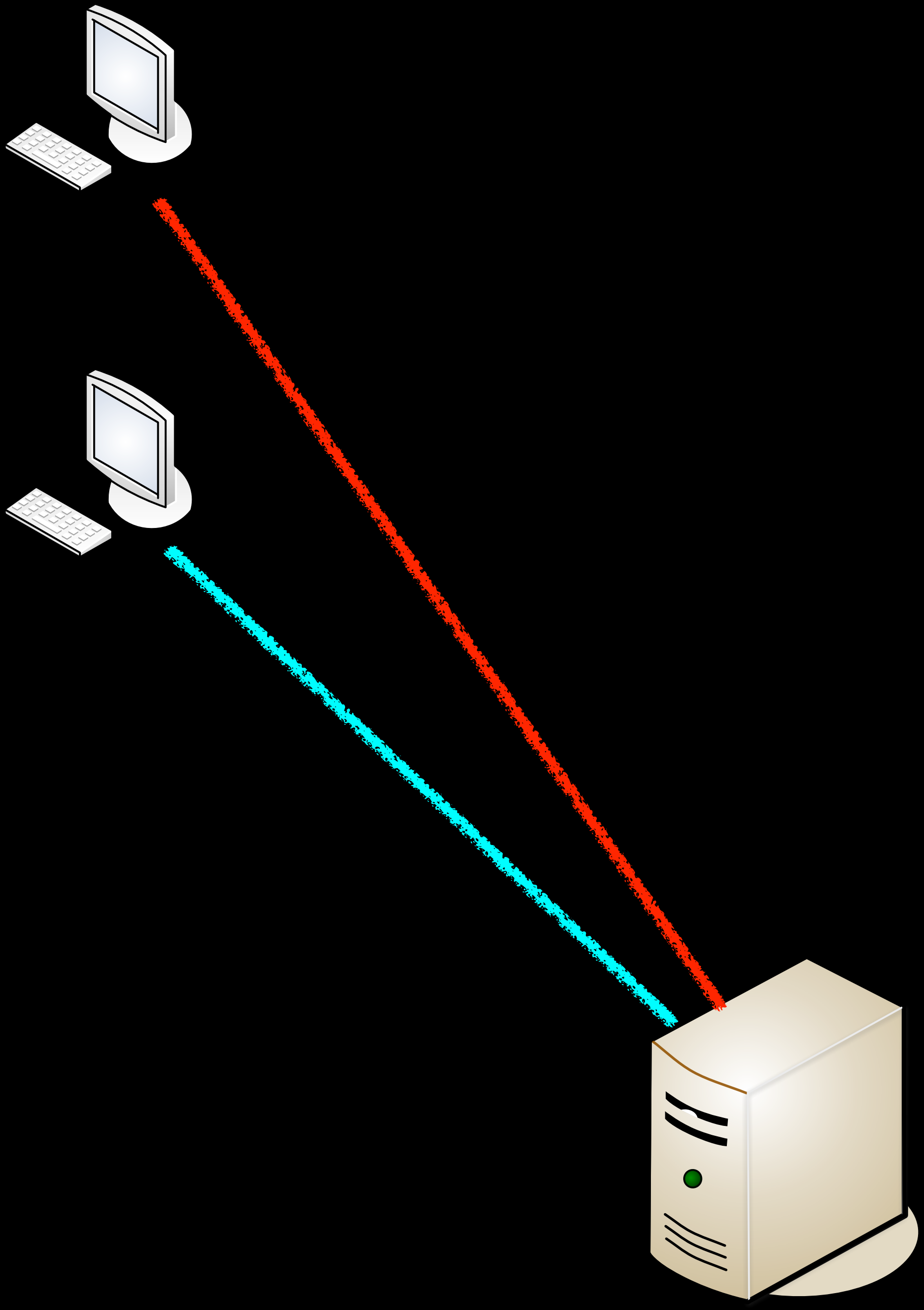
no locking

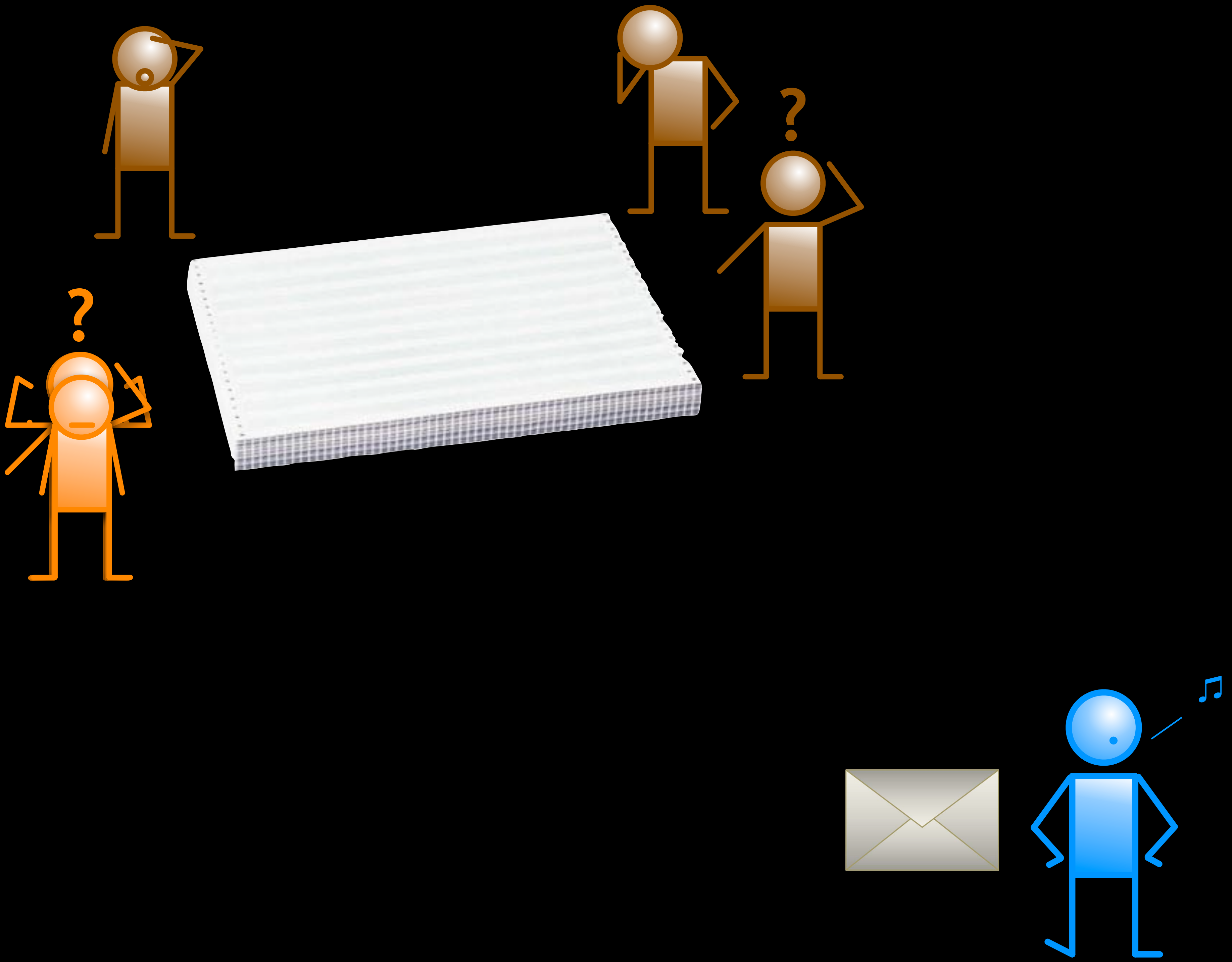
NO LOCKING?!?



NO LOCKING?!?

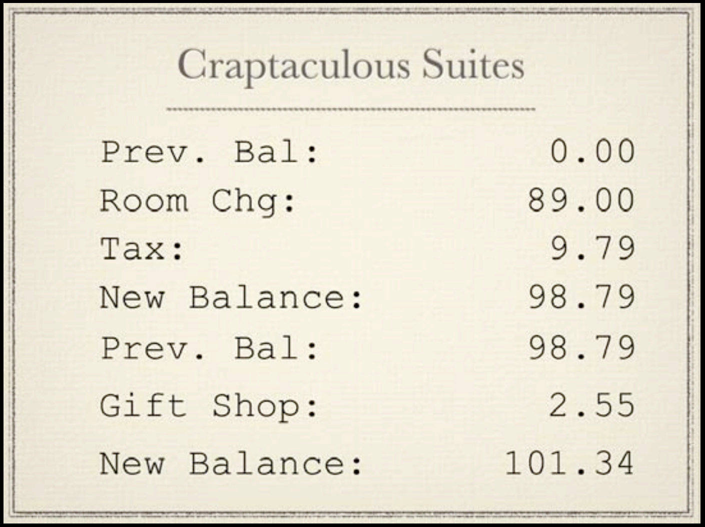






abstraction

distraction



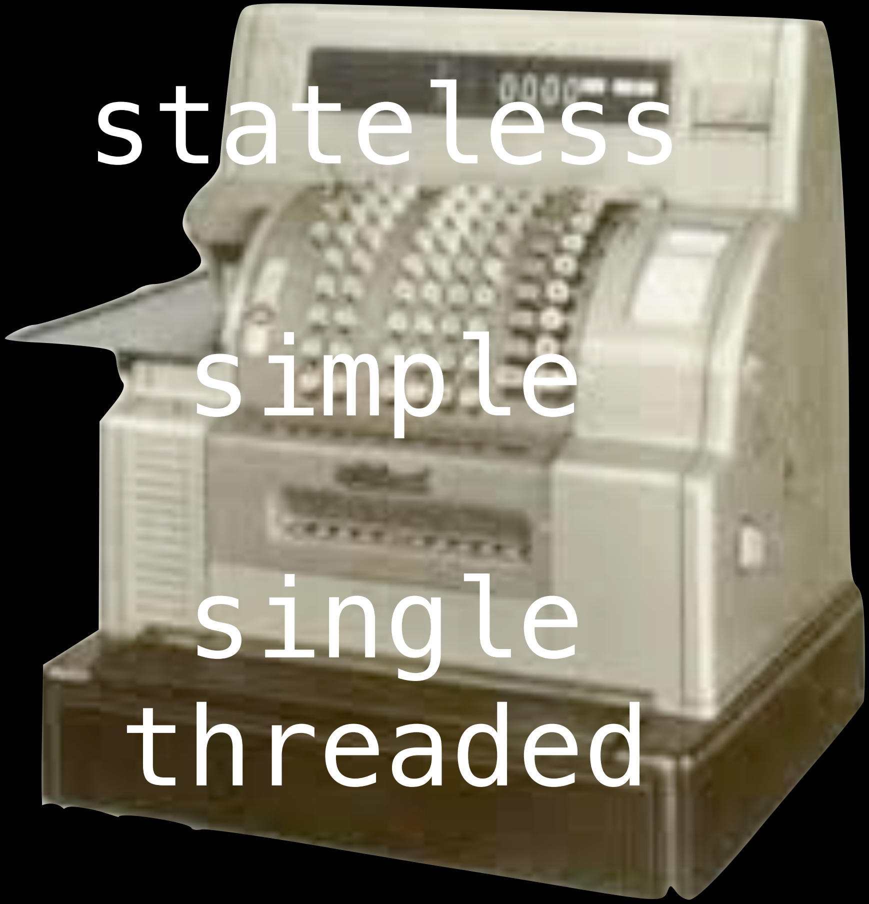
Craptaculous Suites

Prev. Bal:	0.00
Room Chg:	89.00
Tax:	9.79
New Balance:	98.79
Prev. Bal:	98.79
Gift Shop:	2.55
New Balance:	101.34

≠

Lesson #1:

Don't mistake the abstraction for the real thing.



stateless

simple

single threaded

stateful

complex

essential & accidental

concurrency hell

Nikon
D40x
Fn
Nikon
AF-S DX NIKKOR
ED 18-55mm 1:3.5-5.6GII
35
45
55
A
M
M
A
S
P

SETUP
AUD
VCR
TV
POWER
CBL
SELECT
A
B
C
DAY
LAST
FAV
VOL
MUTE
CH
1
2
3
4
5
6
7
8
9
0
HELP
MUSIC/ENTER
REW
PLAY
PAUSE
FFWD
STOP
REC
BYPASS
TV/VCR

Lesson #2:

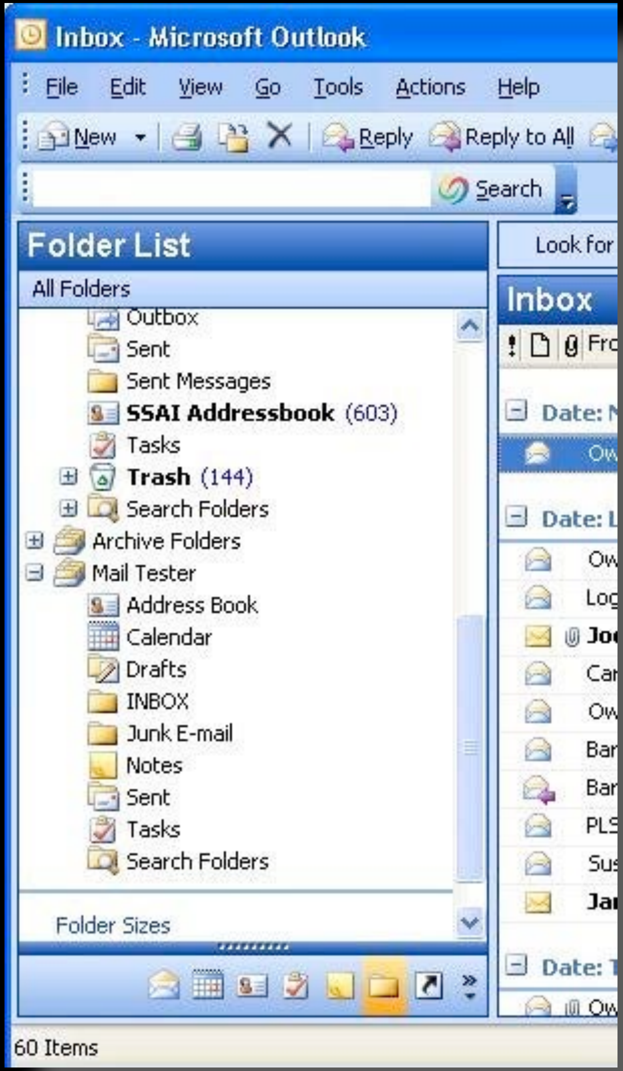
Once internalized, abstractions are hard to shake off.

F1
F2
F3
F4
F5
@ 2
# 3
$ 4
% 5
^ 6
& 7
Q
W
E
R
T
Y
A
S
D
F
G
Z
X
C
V
Alt

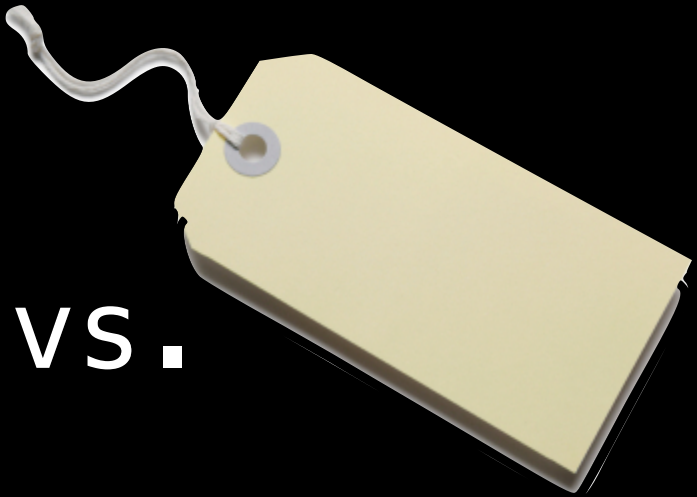
UNDERWOOD

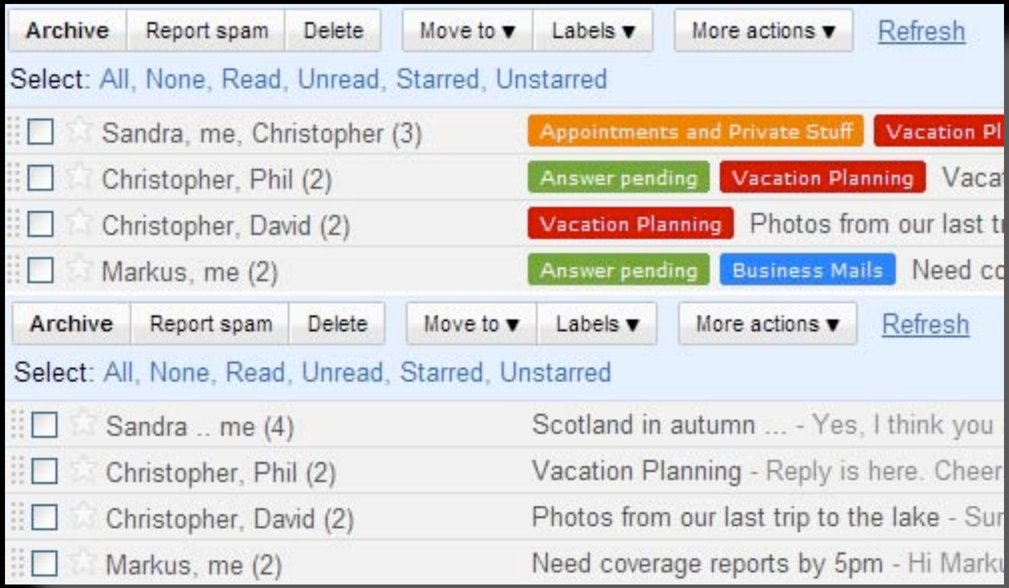
Lesson #3:

Abstractions are both walls & prisons.



vs.




Archive Report spam Delete Move to ▾ Labels ▾ More actions ▾ Refresh
Select: All, None, Read, Unread, Starred, Unstarred

□ ☆ Sandra, me, Christopher (3) Appointments and Private Stuff Vacation Pl

□ ☆ Christopher, Phil (2) Answer pending Vacation Planning Vaca

□ ☆ Christopher, David (2) Vacation Planning Photos from our last t

□ ☆ Markus, me (2) Answer pending Business Mails Need co

Archive Report spam Delete Move to ▾ Labels ▾ More actions ▾ Refresh
Select: All, None, Read, Unread, Starred, Unstarred

□ ☆ Sandra .. me (4) Scotland in autumn ... - Yes, I think you

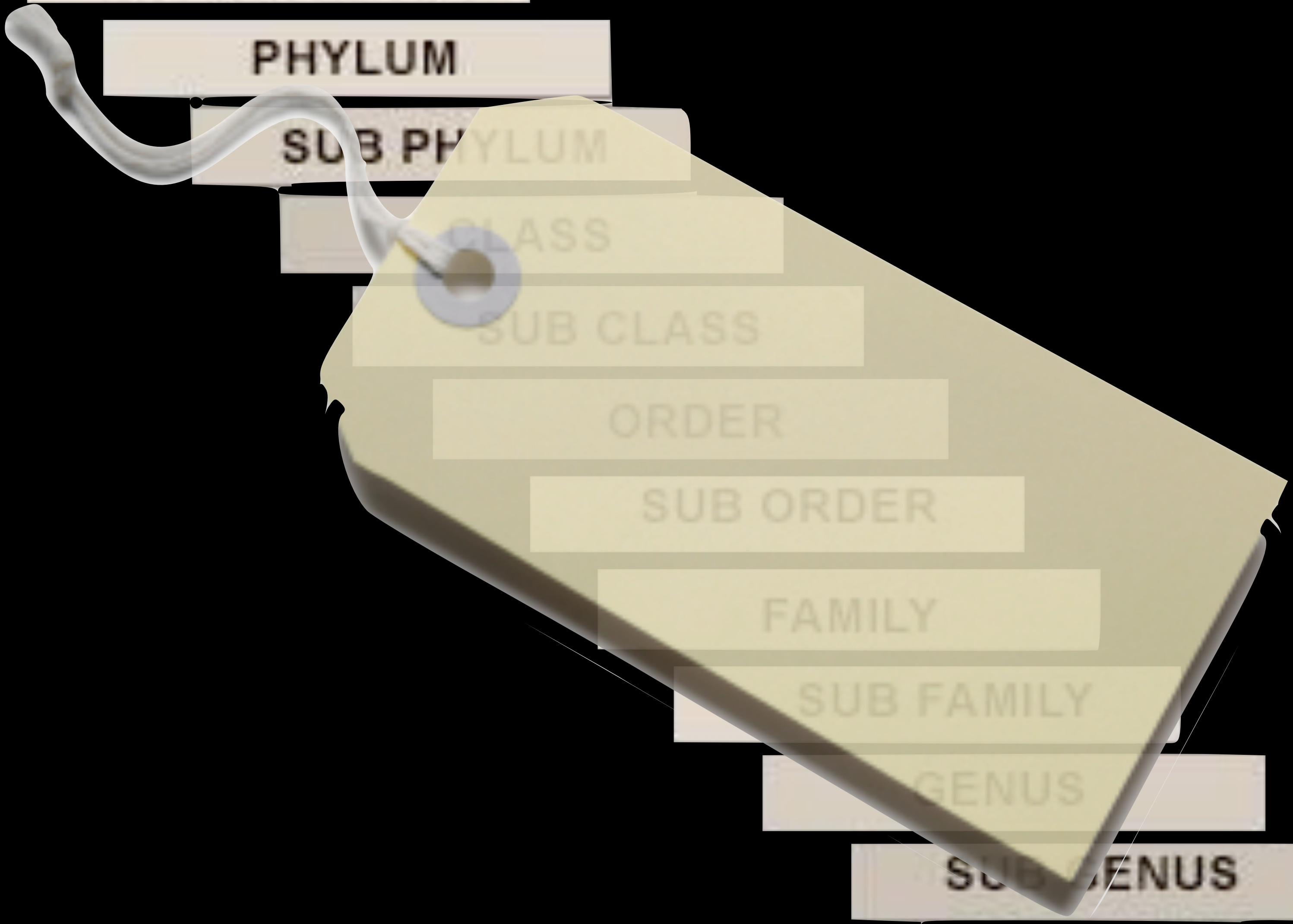
□ ☆ Christopher, Phil (2) Vacation Planning - Reply is here. Cheer

□ ☆ Christopher, David (2) Photos from our last trip to the lake - Sur

□ ☆ Markus, me (2) Need coverage reports by 5pm - Hi Marku

Outlook
break

GMail



KINGDOM

SUBKINGDOM

PHYLUM

SUB PHYLUM

CLASS

SUB CLASS

ORDER

SUB ORDER

FAMILY

SUB FAMILY

GENUS

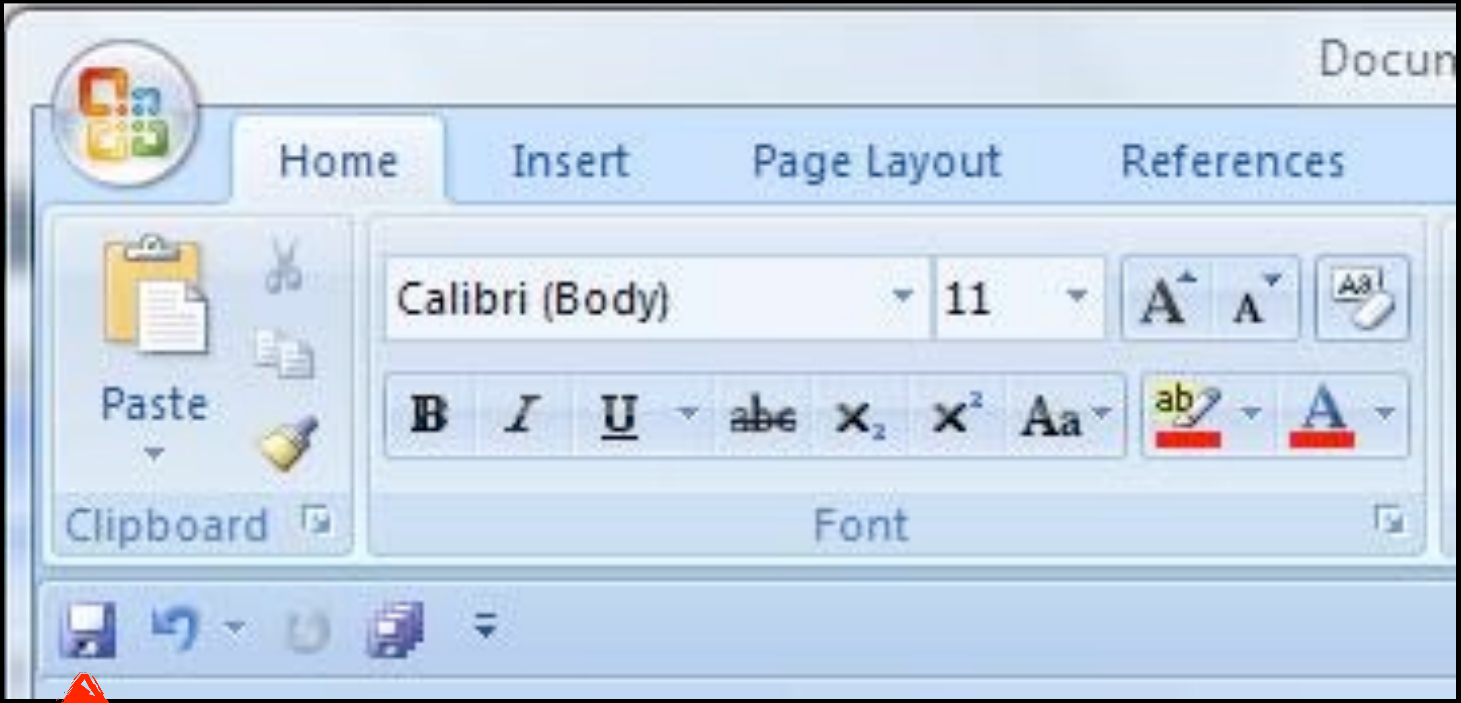
SUB GENUS

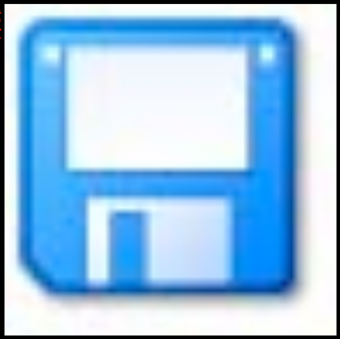
SPECIES

SUB SPECIES

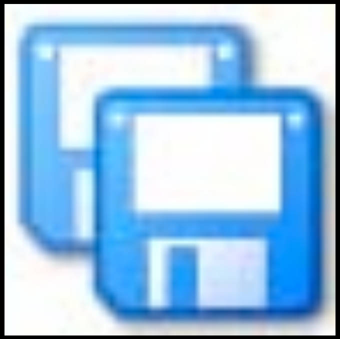
#mammal #layseggs #leftovary

0 0 1 1 0 1 1 0 1
1 0 0 1 0 1 0 1
0 0 1 1 0 1 1 1
1 0 1 0 1 0

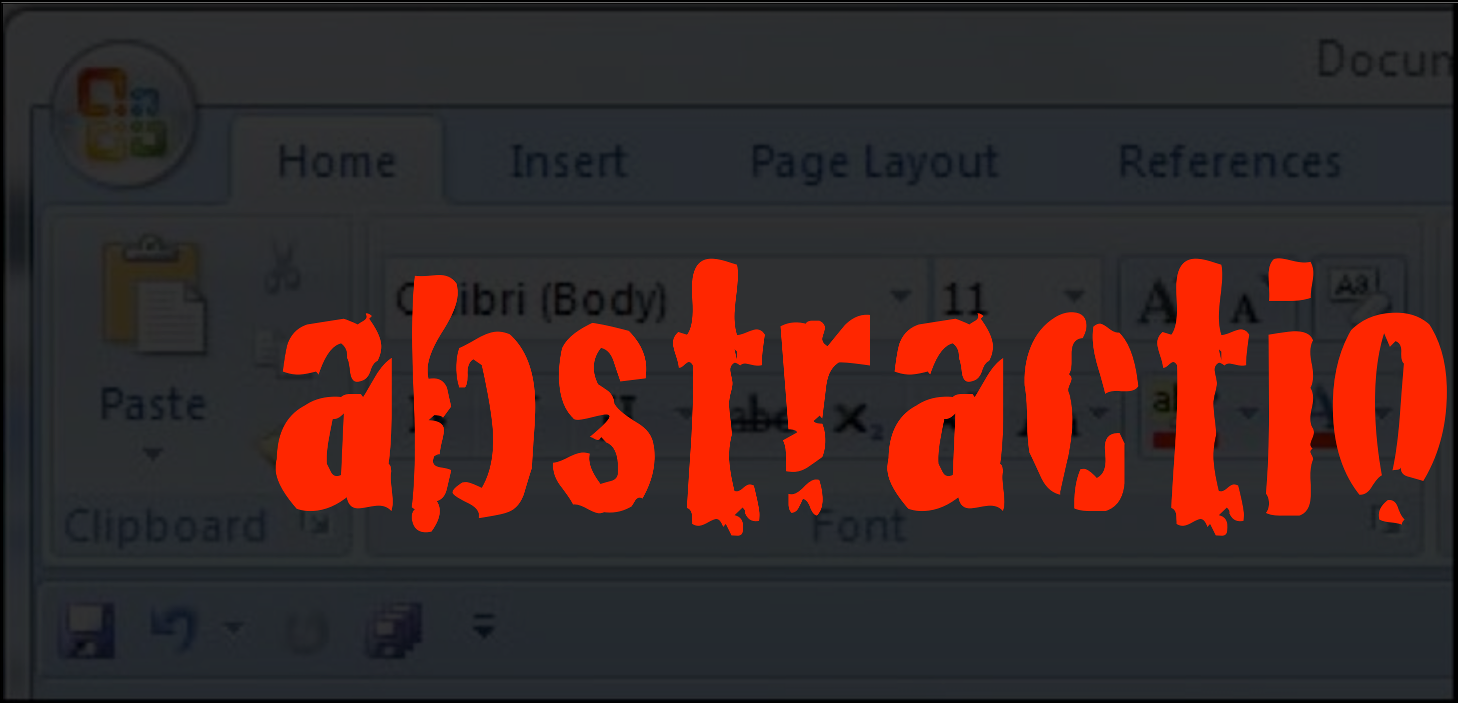

Docun
Home
Insert
Page Layout
References
Calibri (Body)
11
Paste
Clipboard
Font



?!?



?!?



abstraction

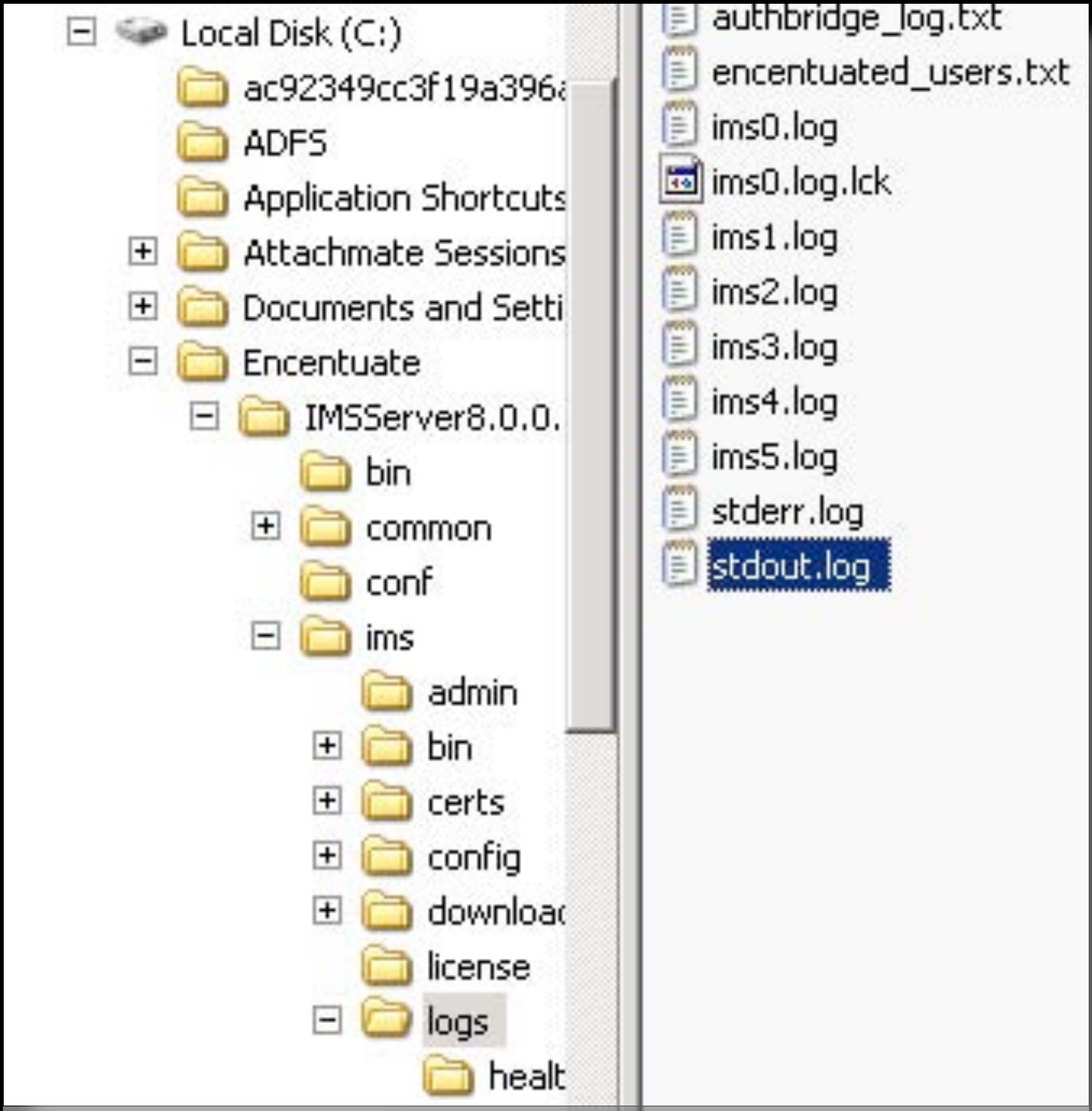
?!?

?!?

distraction

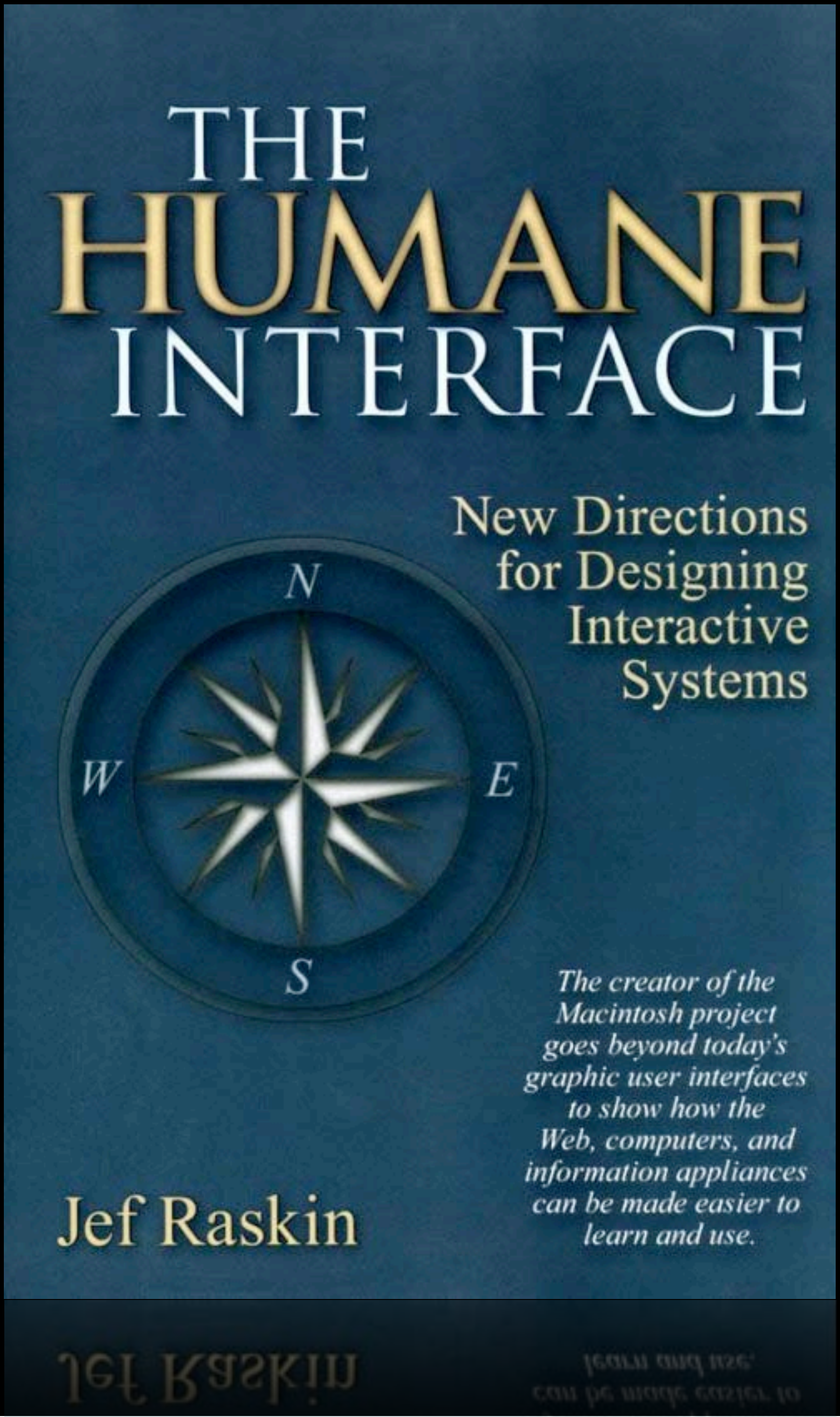
Lesson #4:

Don't name things that expose underlying details.



```
Local Disk (C:)
    ac92349cc3f19a396a
    ADFS
    Application Shortcuts
    Attachmate Sessions
    Documents and Setti
    Encentuate
        IMSServer8.0.0.
            bin
            common
            conf
            ims
                admin
                bin
                certs
                config
                downloac
                license
                logs
                    healt
```

```
authbridge_log.txt
encentuated_users.txt
ims0.log
ims0.log.lck
ims1.log
ims2.log
ims3.log
ims4.log
ims5.log
stderr.log
stdout.log
```

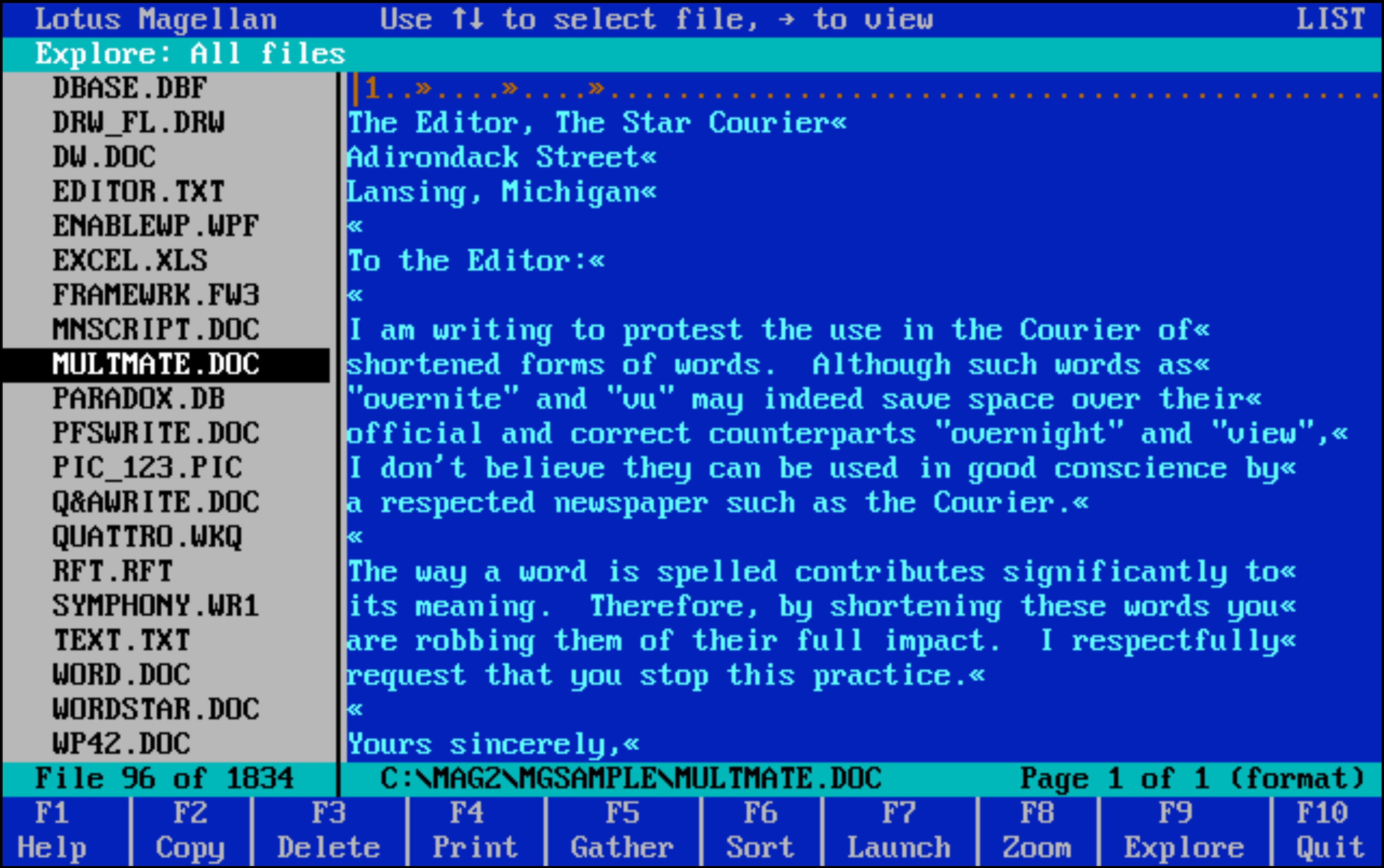


THE HUMANE INTERFACE

New Directions for Designing Interactive Systems

The creator of the Macintosh project goes beyond today's graphic user interfaces to show how the Web, computers, and information appliances can be made easier to learn and use.

Jef Raskin



```
Lotus Magellan      Use ↑↓ to select file, → to view                LIST
Explore: All files
```

```
DBASE.DBF
DRW_FL.DRW
DW.DOC
EDITOR.TXT
ENABLEWP.WPF
EXCEL.XLS
FRAMEWRK.FW3
MNSCRIPT.DOC
MULTMATE.DOC
PARADOX.DB
PFSWRITE.DOC
PIC_123.PIC
Q&AWRITE.DOC
QUATTRO.WKQ
RFT.RFT
SYMPHONY.WR1
TEXT.TXT
WORD.DOC
WORDSTAR.DOC
WP42.DOC
```

```
1..»....»....».....................................................
The Editor, The Star Courier«
Adirondack Street«
Lansing, Michigan«
«
To the Editor:«
«
I am writing to protest the use in the Courier of«
shortened forms of words.  Although such words as«
"overnite" and "vu" may indeed save space over their«
official and correct counterparts "overnight" and "view",«
I don't believe they can be used in good conscience by«
a respected newspaper such as the Courier.«
«
The way a word is spelled contributes significantly to«
its meaning.  Therefore, by shortening these words you«
are robbing them of their full impact.  I respectfully«
request that you stop this practice.«
«
Yours sincerely,«
```

```
File 96 of 1834   C:\MAG2\MGSAMPLE\MULTMATE.DOC          Page 1 of 1 (format)
```

Lesson #3:

Abstractions are both walls & prisons.

iPad
9:41 AM
27
Calendar
Contacts
Notes
Maps
Videos
YouTube
iTunes
App Store
Settings
7
Safari
Mail
Photos
iPod

27
Photos
iPod
*
Local Disk (C:)
ac02549cc3f19a396
ADFS
Application Shortcuts
Attachmate Sessions
Documents and Setti
Encentuate
IMSServer8.0.0.
bin
common
conf
ims
admin
bin
certs
config
download
license
logs
healt
authbridge_log.txt
encentuated_users.txt
ims0.log
ims0.log.lck
ims1.log
ims2.log
ims3.log
ims4.log
ims5.log
stderr.log
stdout.log

Lesson #2:

Once internalized, abstractions are hard to shake off

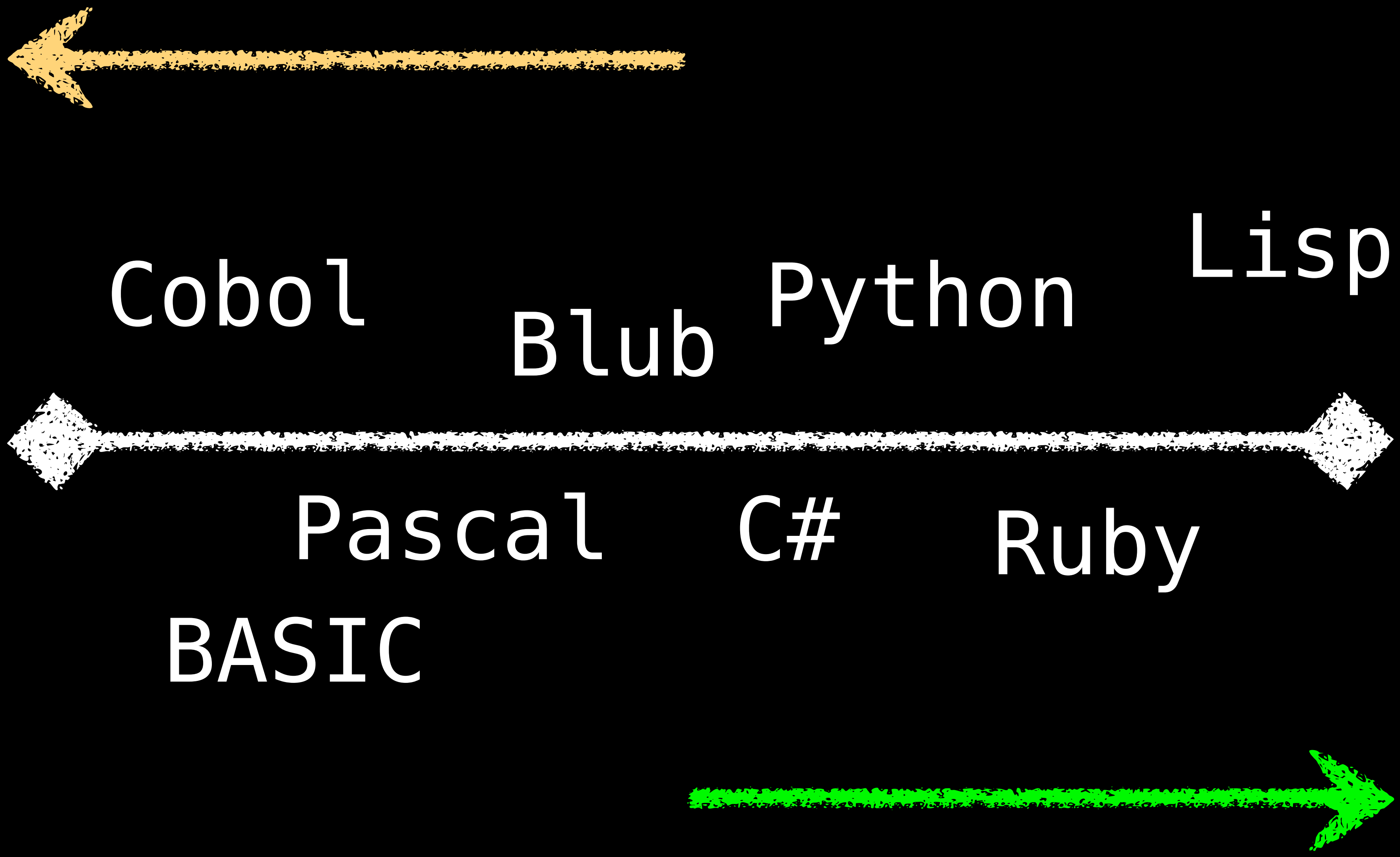
suck/rock dichotomy

rocks! sucks!

rocks! sucks!

rocks! sucks!

blub paradox



Lisp

Cobol

Python

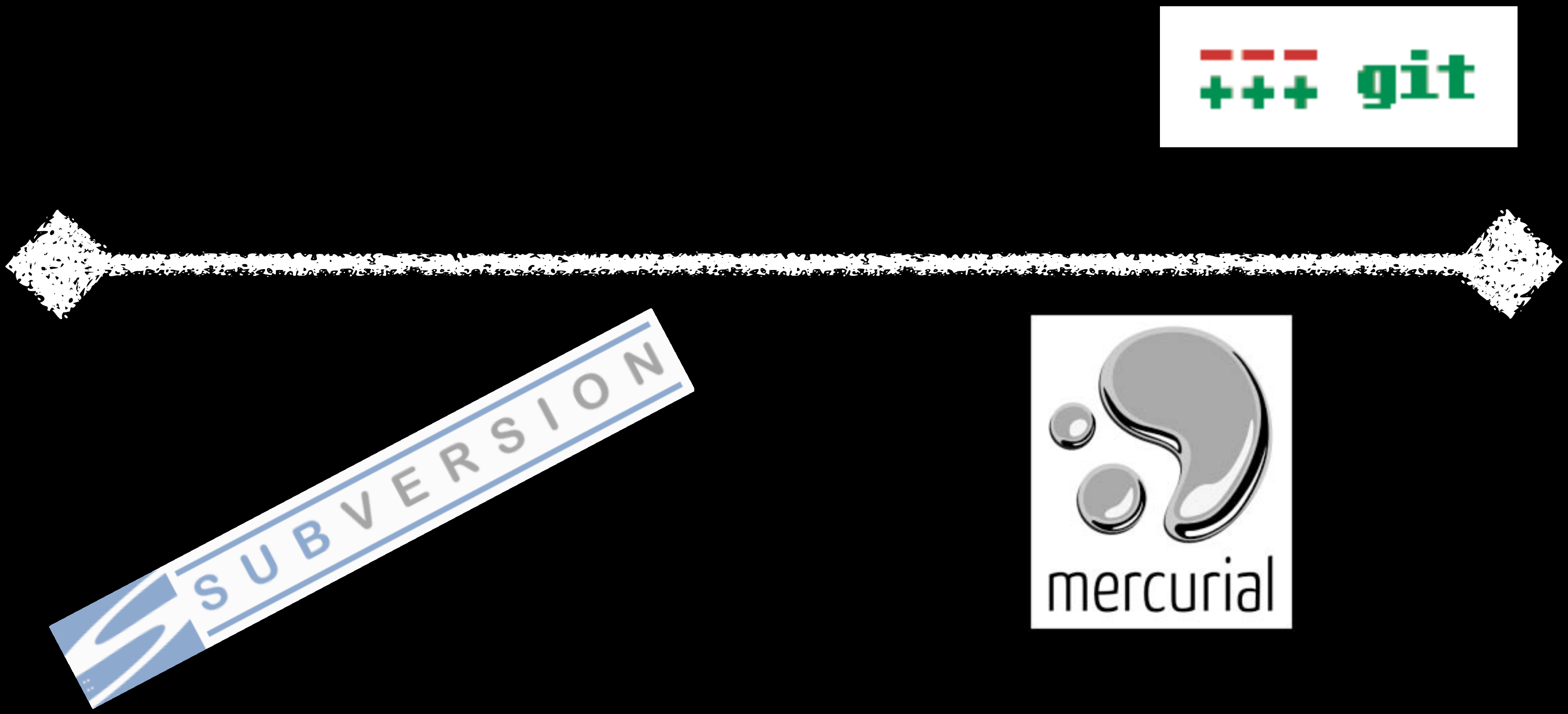
Blub

Pascal

C#

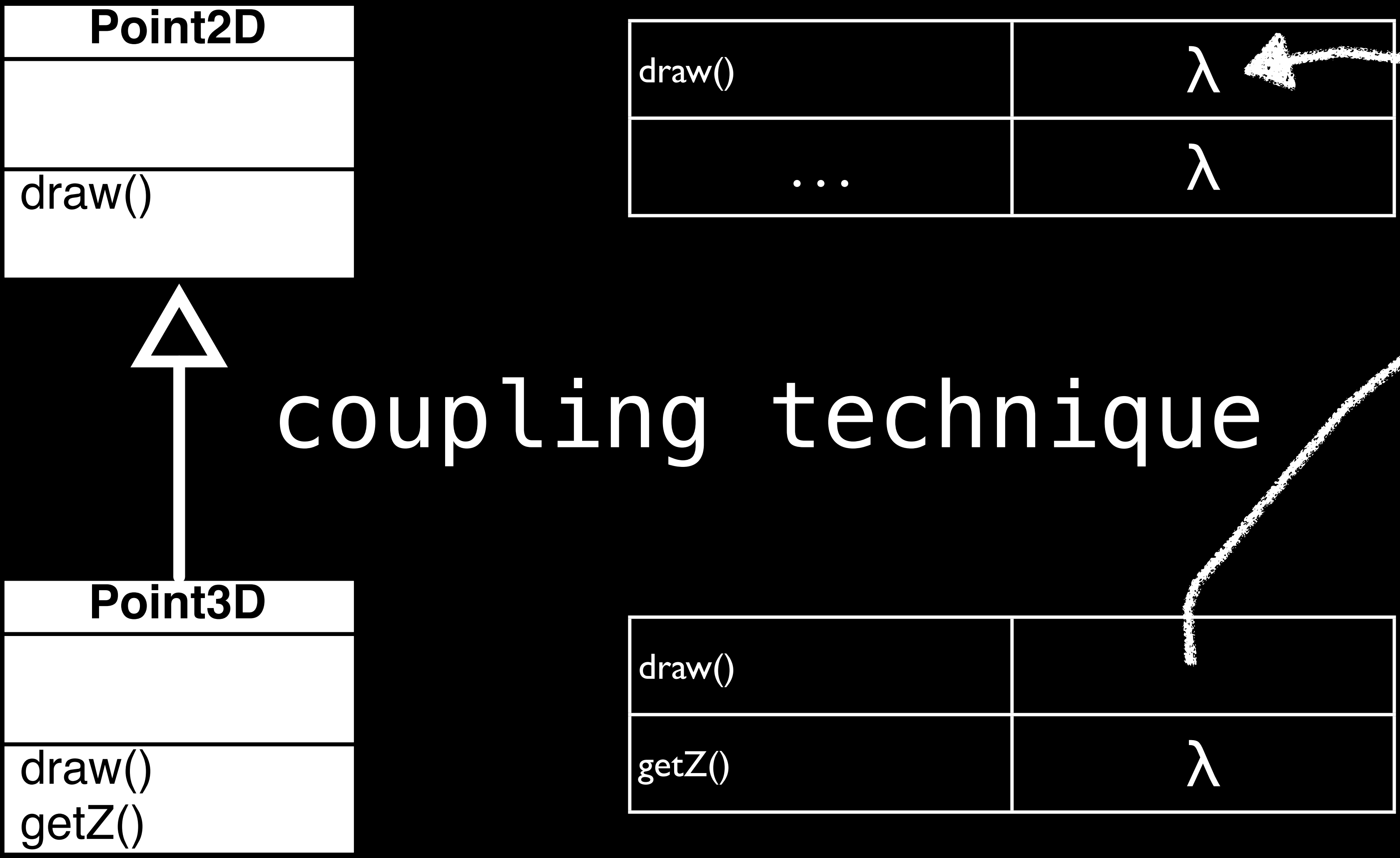
Ruby

BASIC


git
SUBVERSION
mercurial

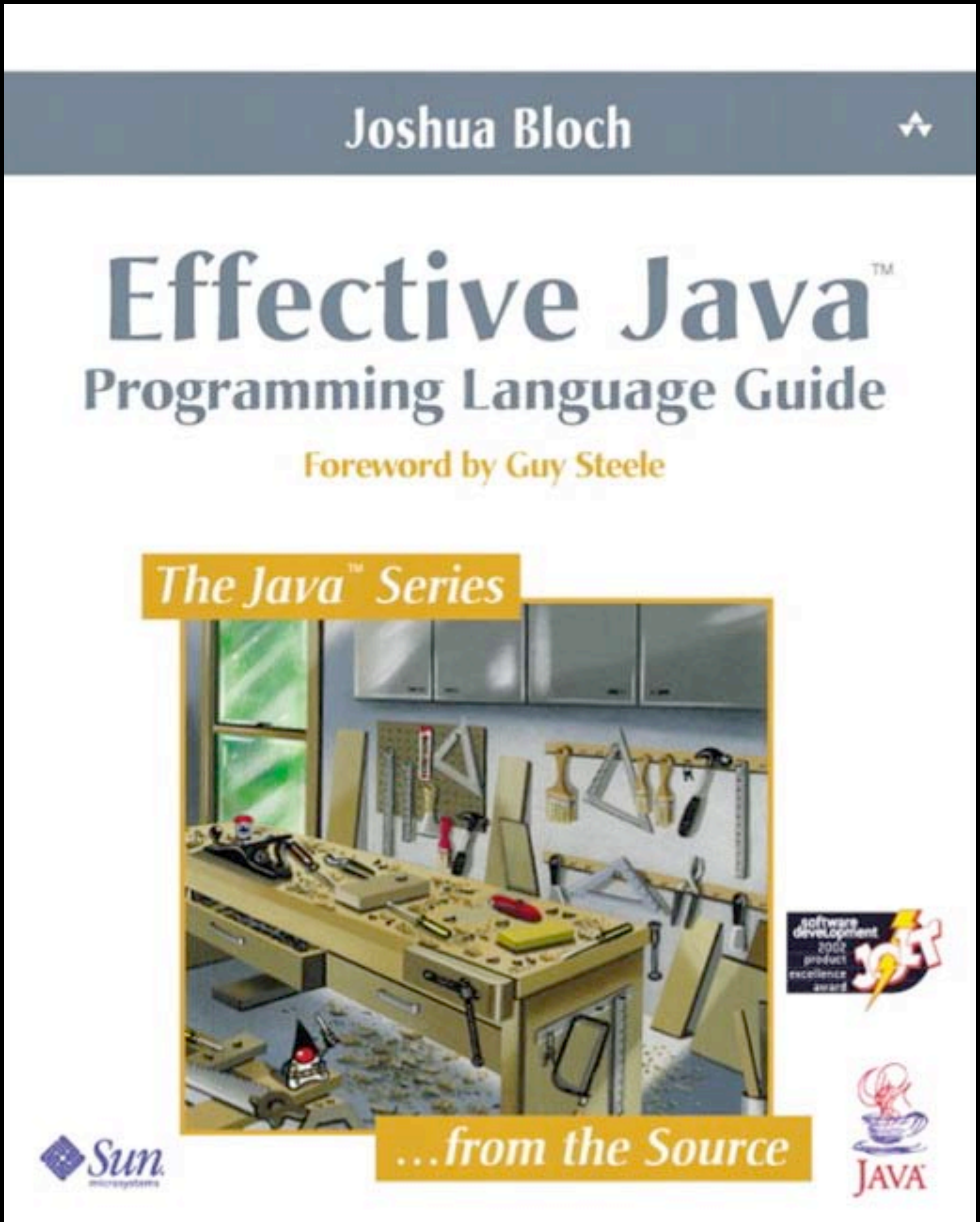
language abstractions

coupling technique



draw()	λ
…	λ

draw()	
getZ()	λ



Joshua Bloch

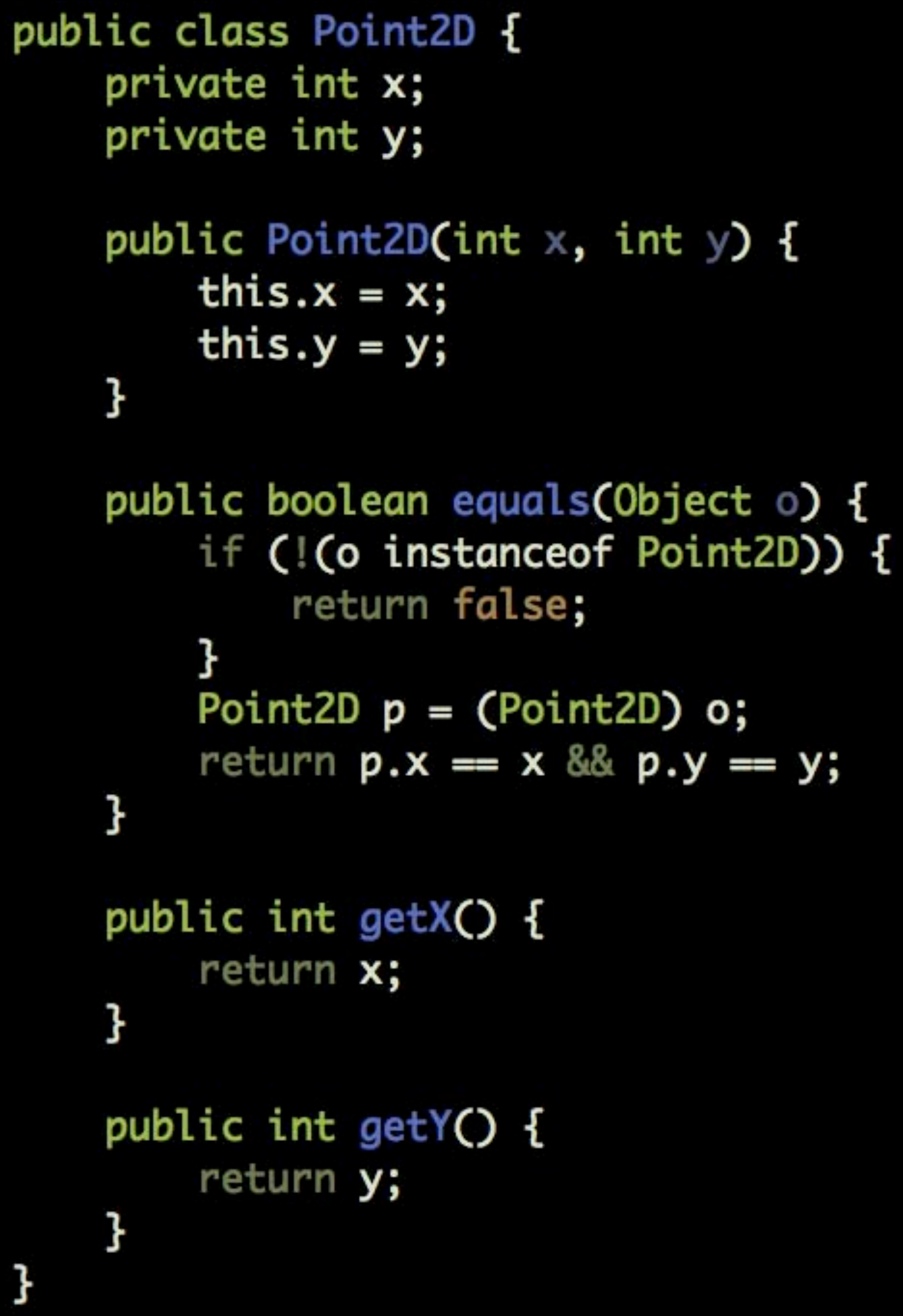
Effective Java™

Programming Language Guide

Foreword by Guy Steele

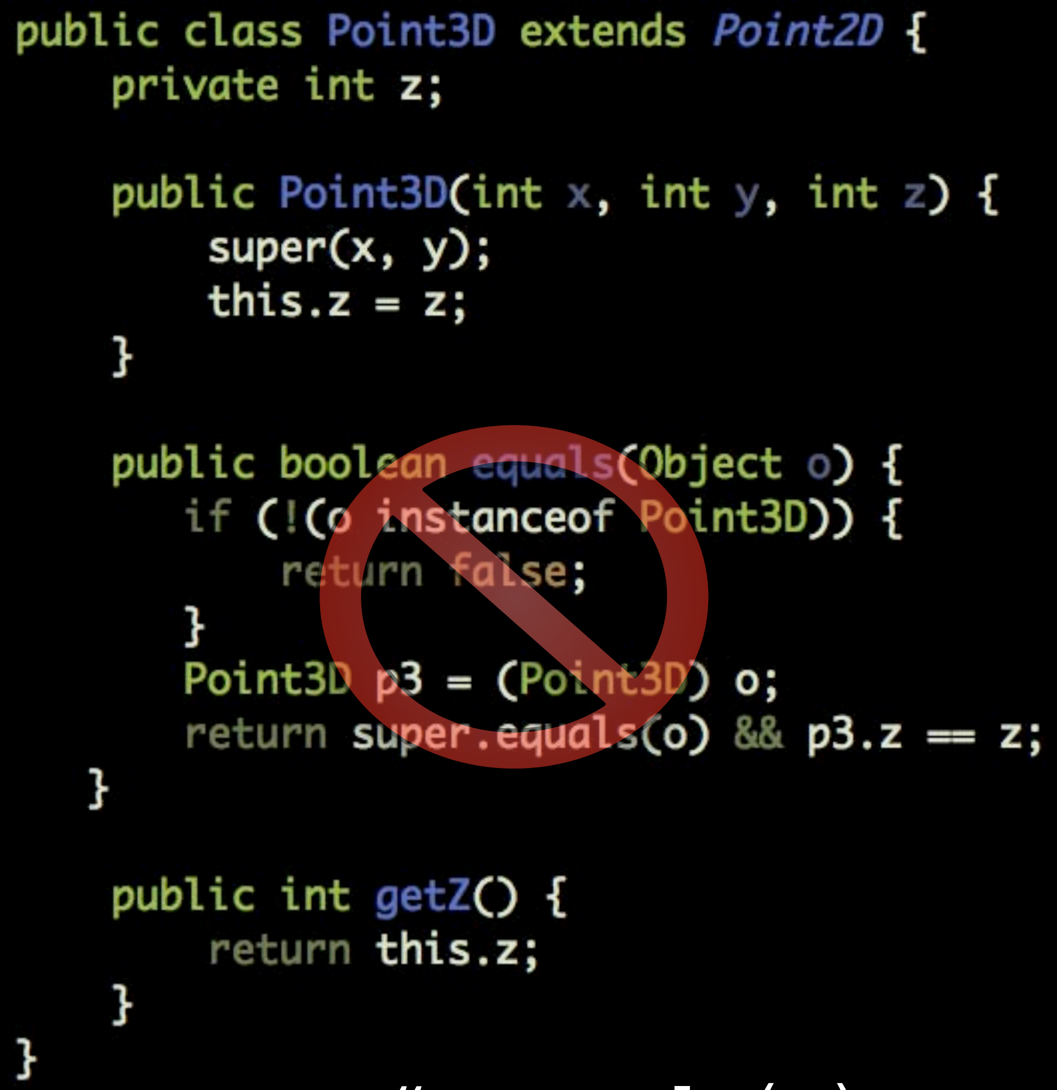
The Java™ Series

...from the Source



```
public class Point2D {
    private int x;
    private int y;

    public Point2D(int x, int y) {
        this.x = x;
        this.y = y;
    }

    public boolean equals(Object o) {
        if (!(o instanceof Point2D)) {
            return false;
        }
        Point2D p = (Point2D) o;
        return p.x == x && p.y == y;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }
}
```

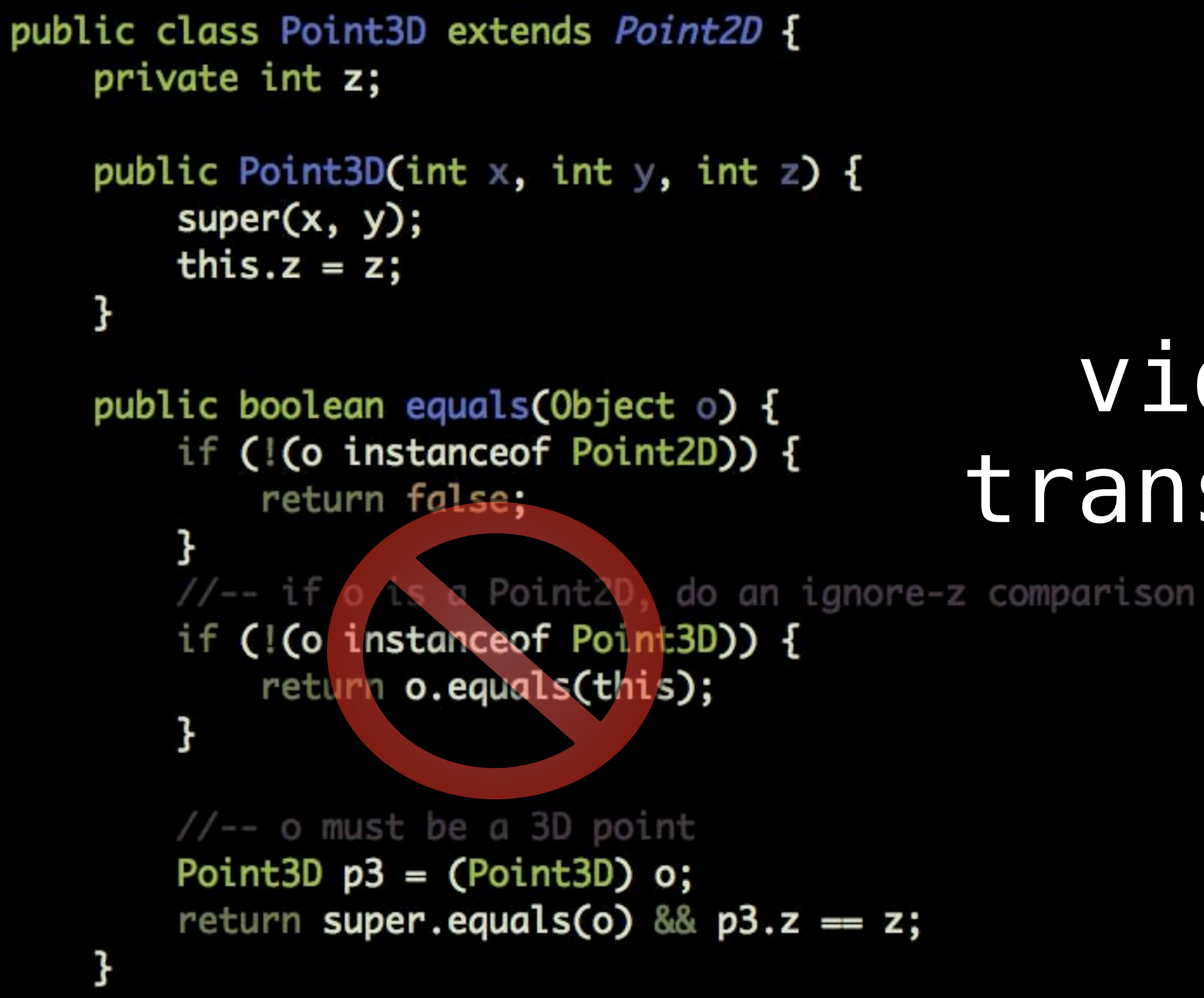
instance check
typecast
equality check



```
public class Point3D extends Point2D {
    private int z;

    public Point3D(int x, int y, int z) {
        super(x, y);
        this.z = z;
    }

    public boolean equals(Object o) {
       if (!(o instanceof Point3D)) {
           return false;
       }
       Point3D p3 = (Point3D) o;
       return super.equals(o) && p3.z == z;
   }

    public int getZ() {
        return this.z;
    }
}
```

violates symmetry

“x.equals(y)must return true if and only if y.equals(x) returns true”

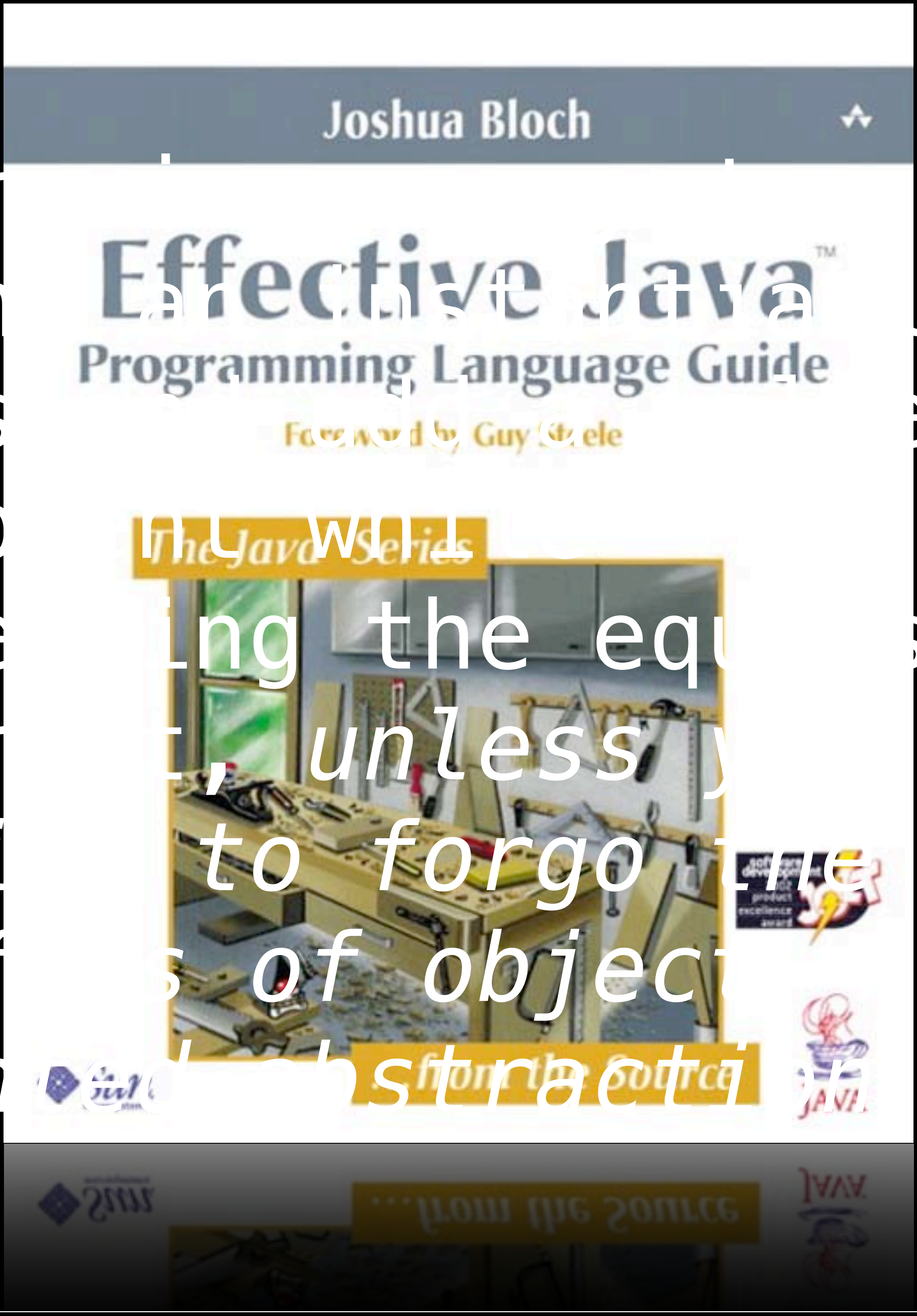


```java
public class Point3D extends Point2D {
    private int z;

    public Point3D(int x, int y, int z) {
        super(x, y);
        this.z = z;
    }

    public boolean equals(Object o) {
        if (!(o instanceof Point2D)) {
            return false;
        }
        //-- if o is a Point2D, do an ignore-z comparison
        if (!(o instanceof Point3D)) {
            return o.equals(this);
        }

        //-- o must be a 3D point
        Point3D p3 = (Point3D) o;
        return super.equals(o) && p3.z == z;
    }
```

violates transitivity

“if x.equals(y)returns true and y.equals(z) returns true, then x.equals(z) must return true”

“There is no way to extend an instantiable class and add a value component while preserving the equals contract, *unless you are willing to forgo the benefits of object-oriented abstraction.*”



object-oriented
programming

object-oriented
programming

abstraction

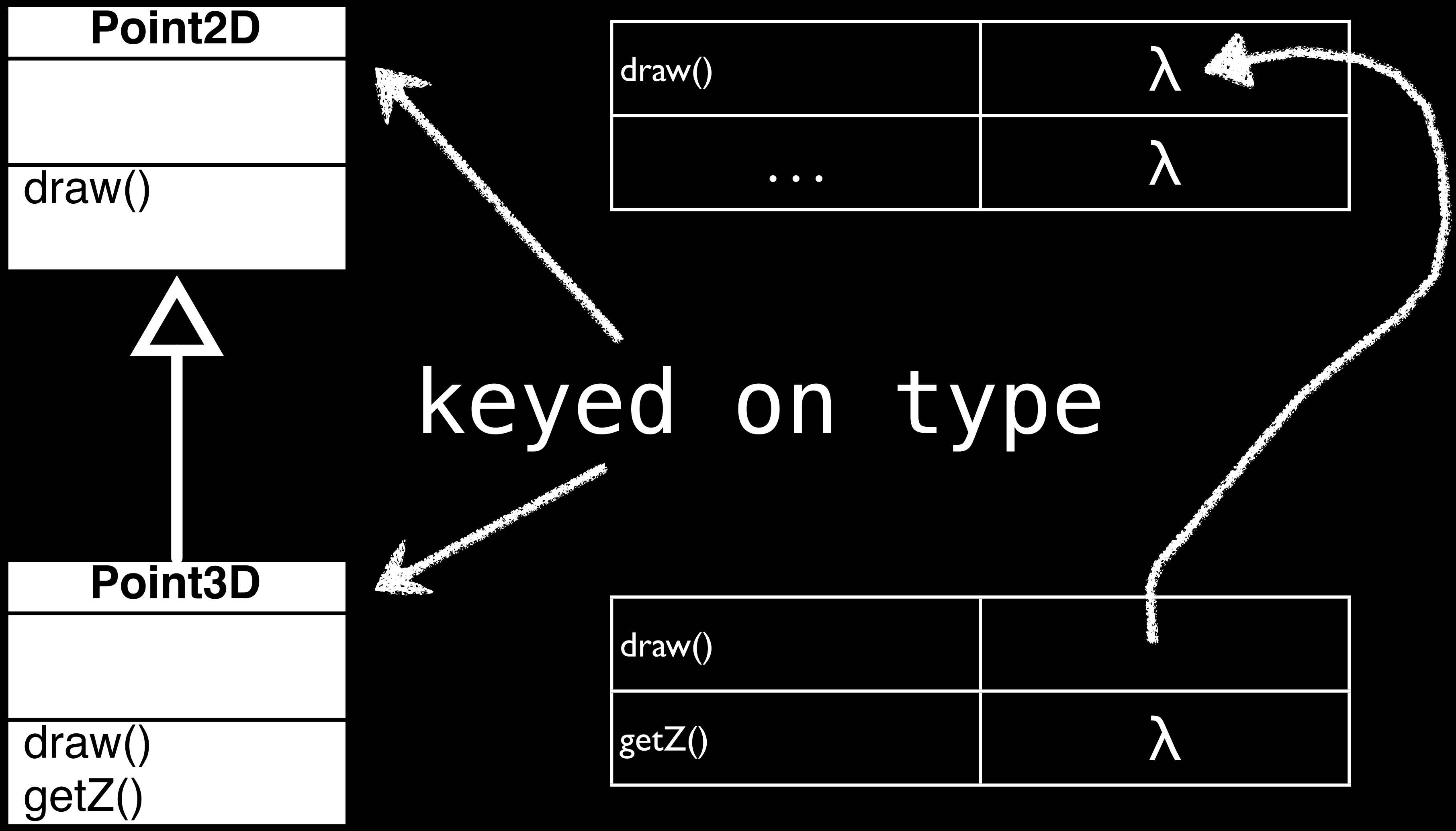
object- oriented

progr amming

distraction

Lesson #5:

Your abstraction isn't perfect.



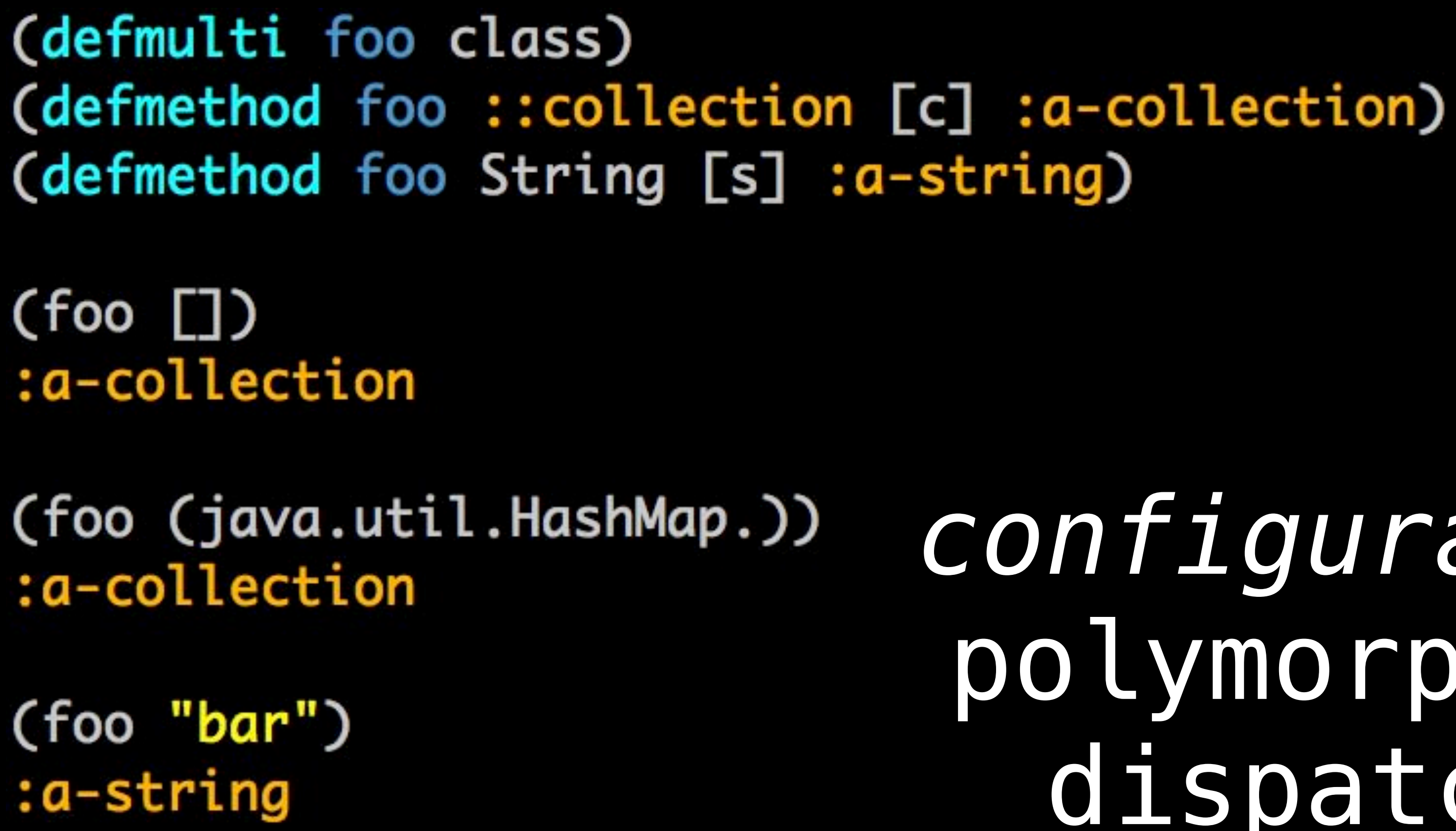
draw()	λ
…	λ

keyed on type

draw()	
getZ()	λ

polymorphic dispatch

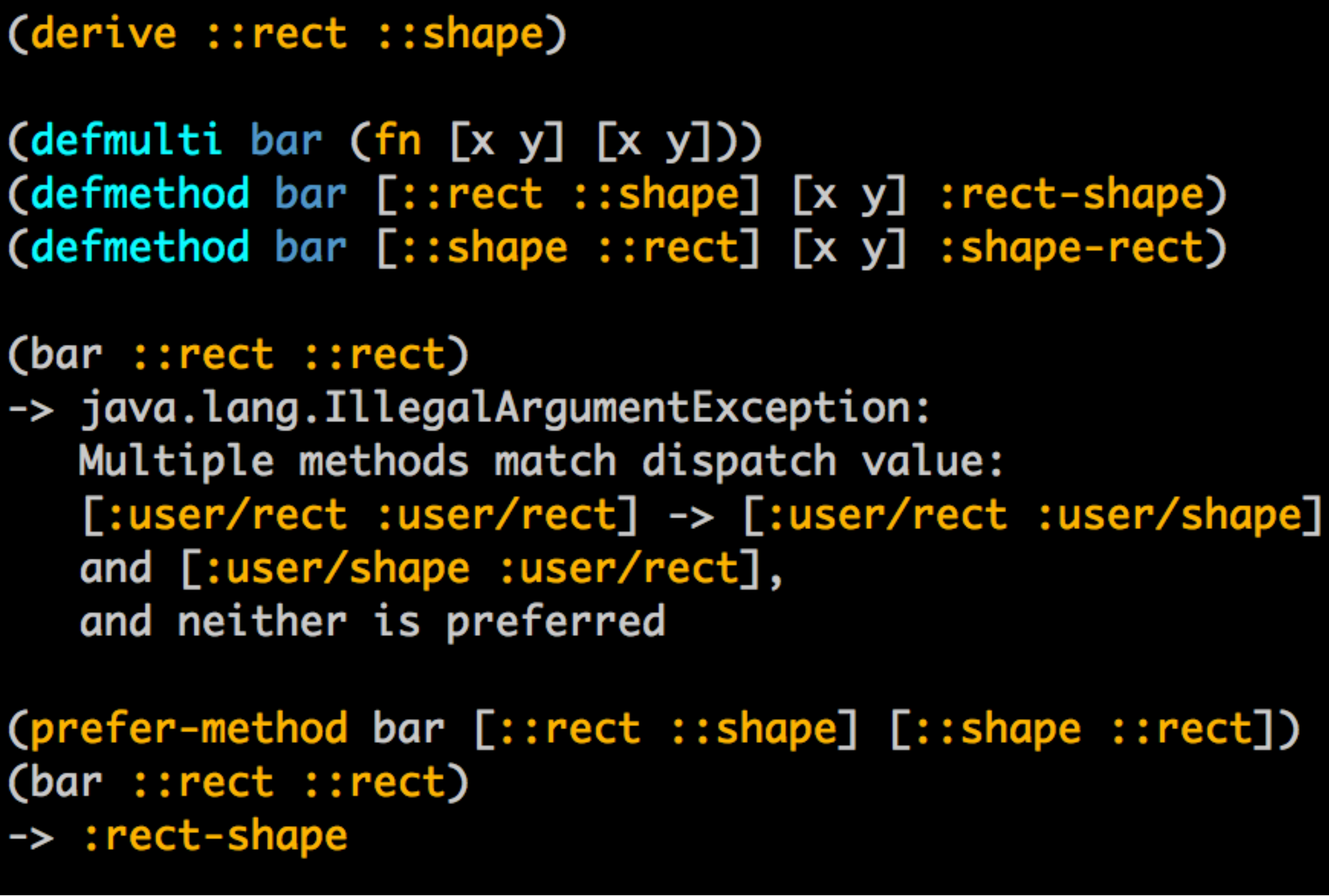
clojure multimethods



```
(defmulti foo class)
(defmethod foo ::collection [c] :a-collection)
(defmethod foo String [s] :a-string)

(foo [])
:a-collection

(foo (java.util.HashMap.))
:a-collection

(foo "bar")
:a-string
```

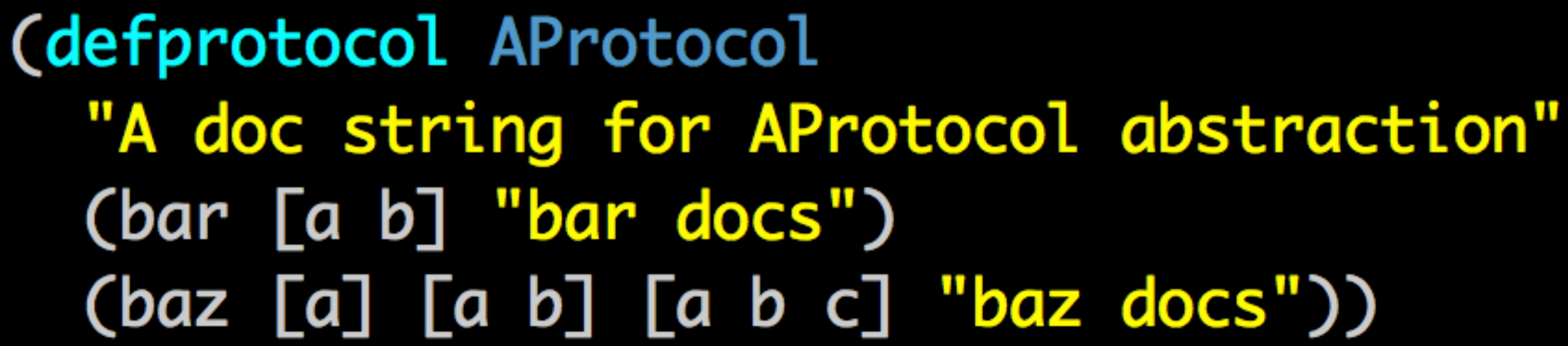
configurable
polymorphic
dispatch

disambiguation



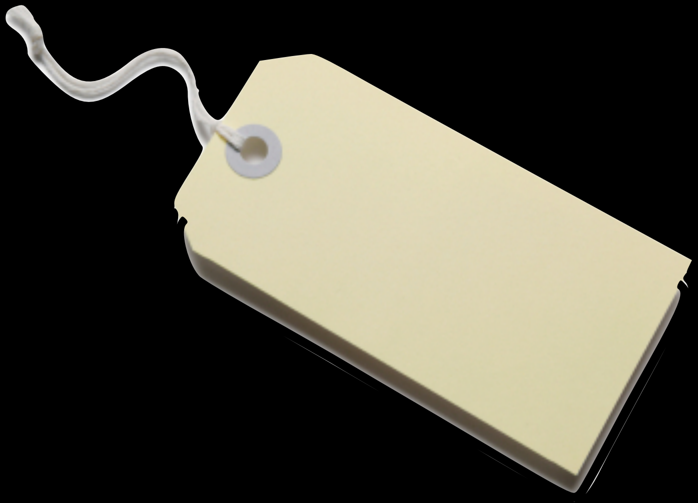
```
(derive ::rect ::shape)

(defmulti bar (fn [x y] [x y]))
(defmethod bar [::rect ::shape] [x y] :rect-shape)
(defmethod bar [::shape ::rect] [x y] :shape-rect)

(bar ::rect ::rect)
-> java.lang.IllegalArgumentException:
   Multiple methods match dispatch value:
   [:user/rect :user/rect] -> [:user/rect :user/shape]
   and [:user/shape :user/rect],
   and neither is preferred

(prefer-method bar [::rect ::shape] [::shape ::rect])
(bar ::rect ::rect)
-> :rect-shape
```

protocols



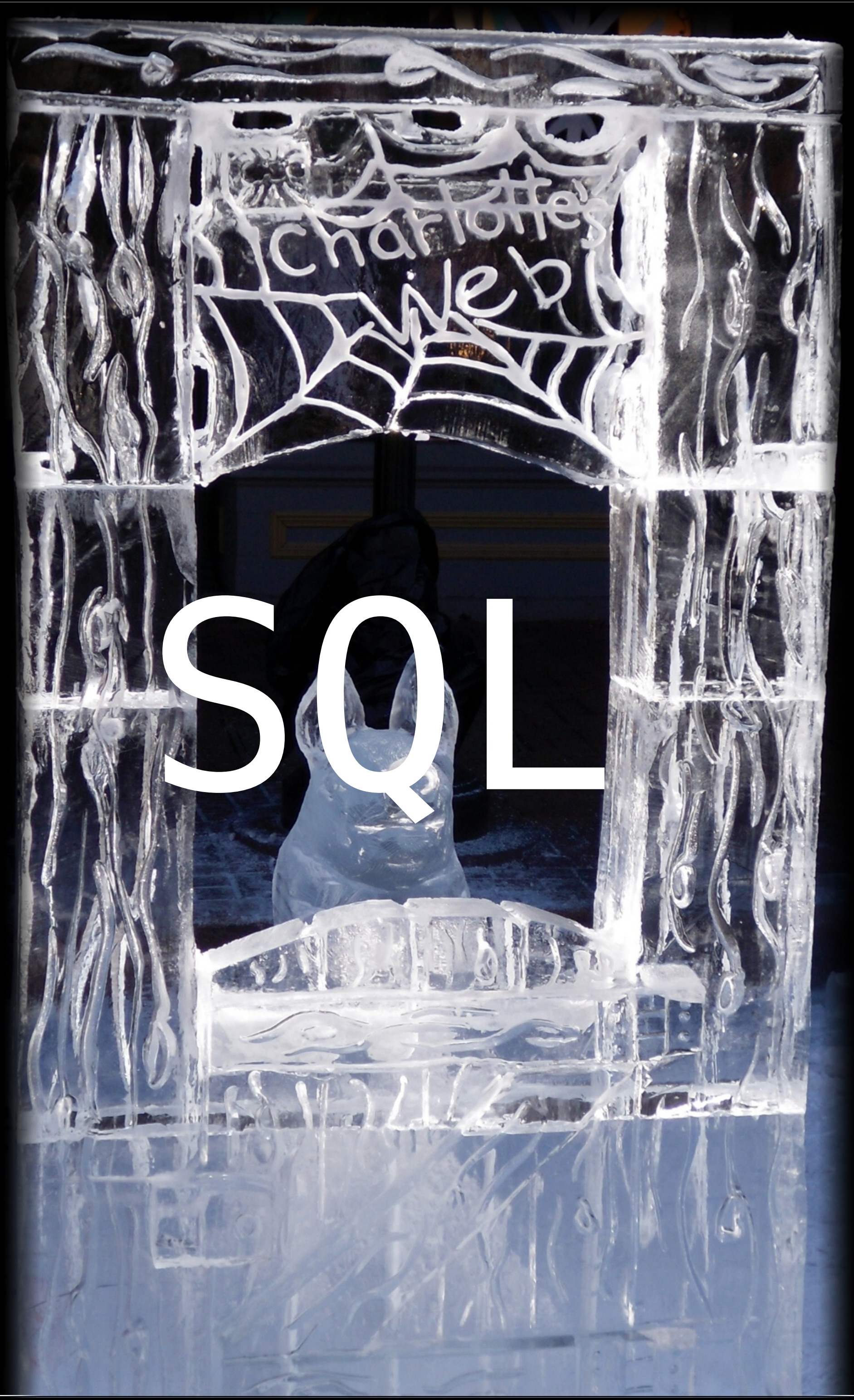
```
(defprotocol AProtocol
  "A doc string for AProtocol abstraction"
  (bar [a b] "bar docs")
  (baz [a] [a b] [a b c] "baz docs"))
```

inheritance



mixin

rigidity



SQL

no SQL

SOA

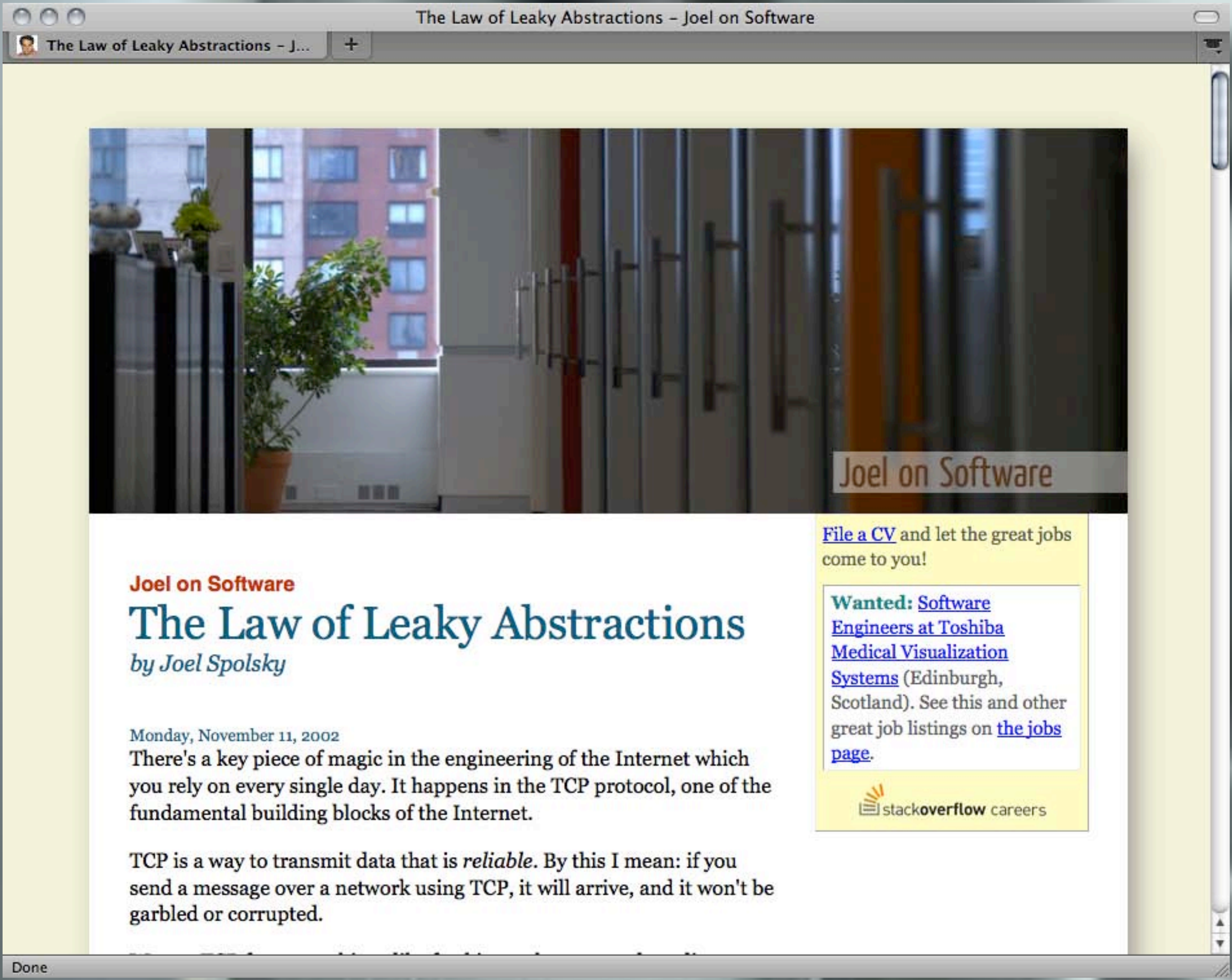
REST

abstraction

distraction

Lesson #6:

Understand the implications of rigidity


The Law of Leaky Abstractions – Joel on Software
The Law of Leaky Abstractions – J...
+
Joel on Software
Done

Joel on Software

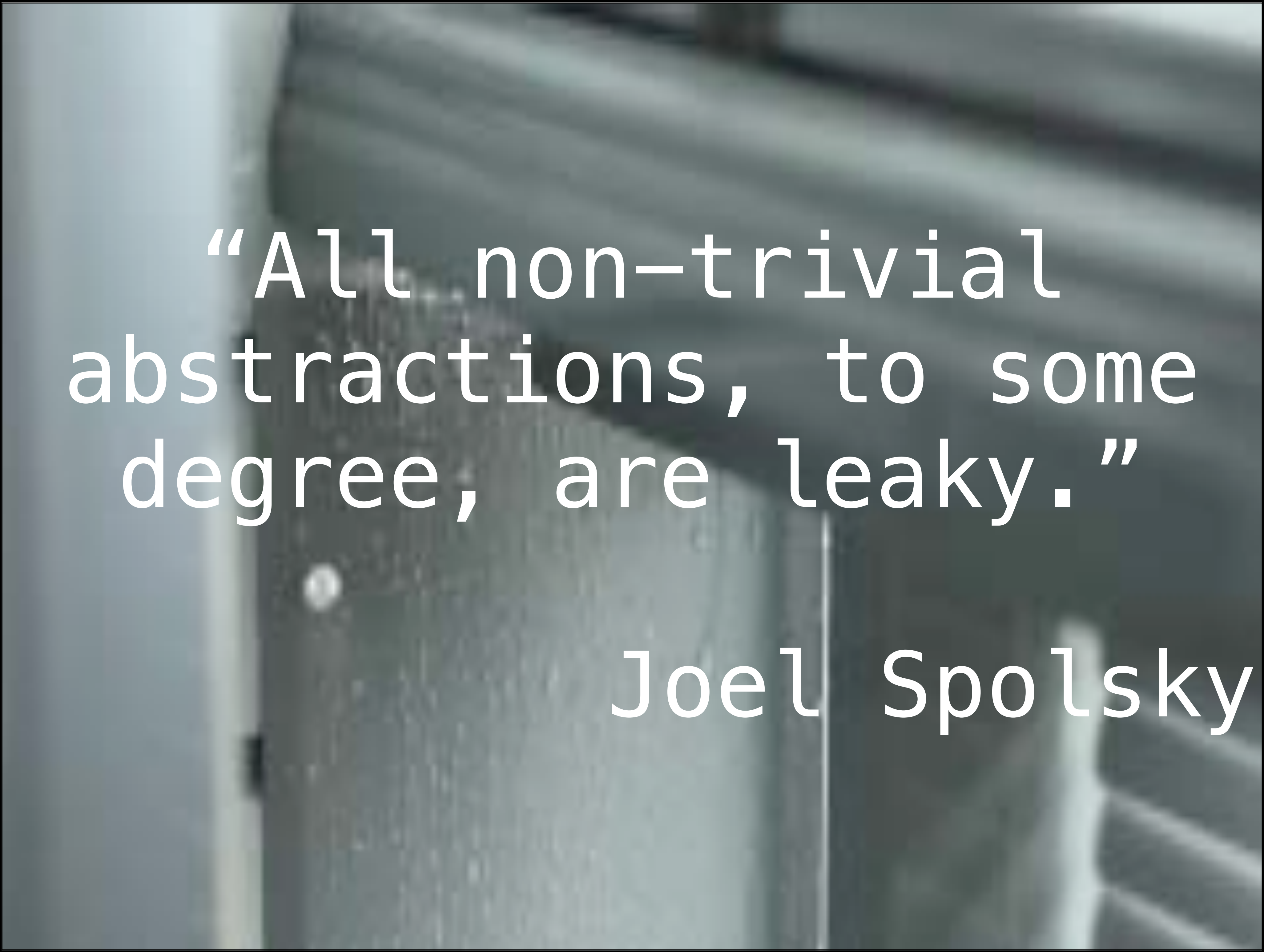
The Law of Leaky Abstractions

by Joel Spolsky

Monday, November 11, 2002

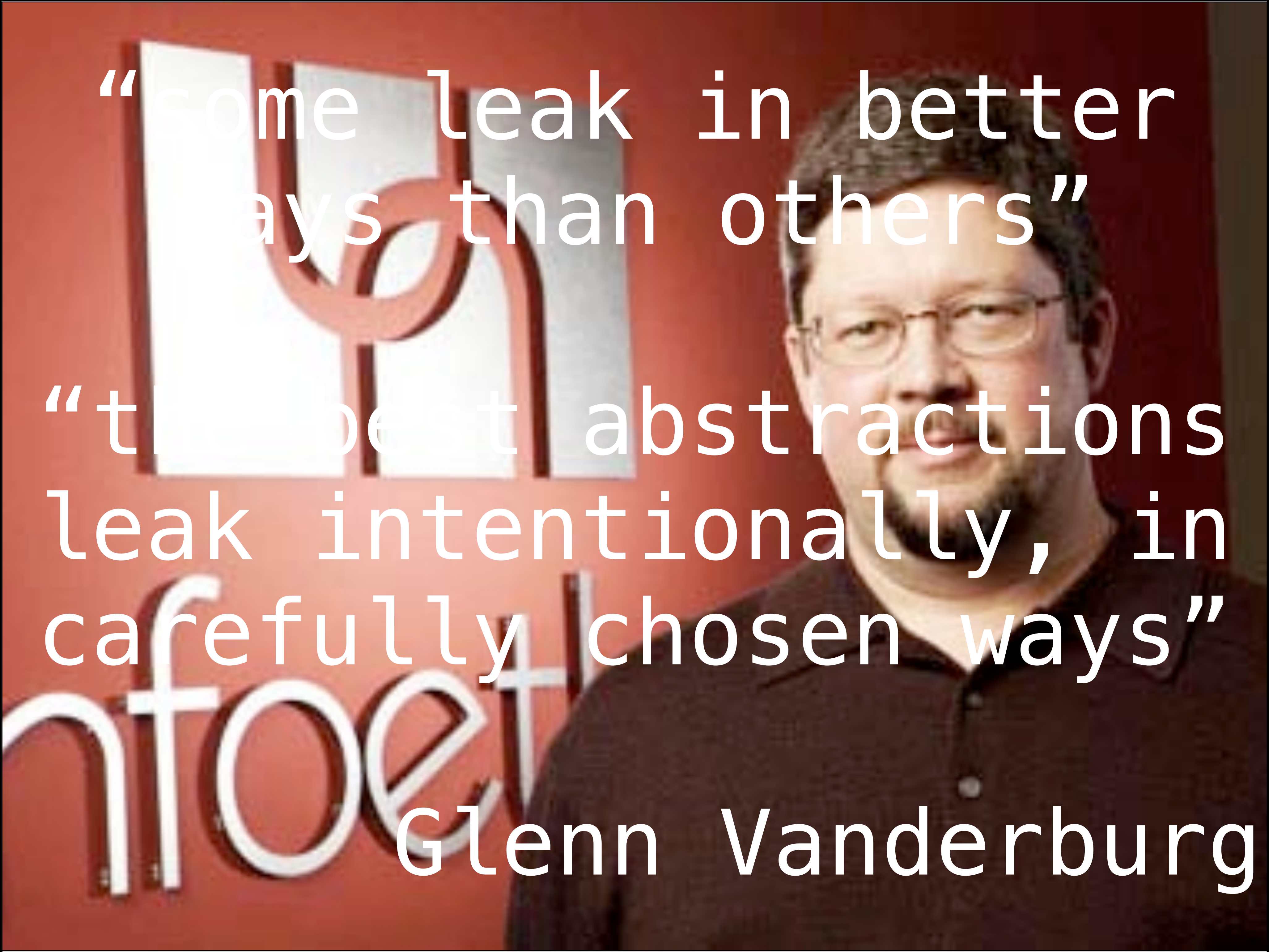
There's a key piece of magic in the engineering of the Internet which you rely on every single day. It happens in the TCP protocol, one of the fundamental building blocks of the Internet.

TCP is a way to transmit data that is *reliable*. By this I mean: if you send a message over a network using TCP, it will arrive, and it won't be garbled or corrupted.



“All non–trivial abstractions, to some degree, are leaky.”

Joel Spolsky

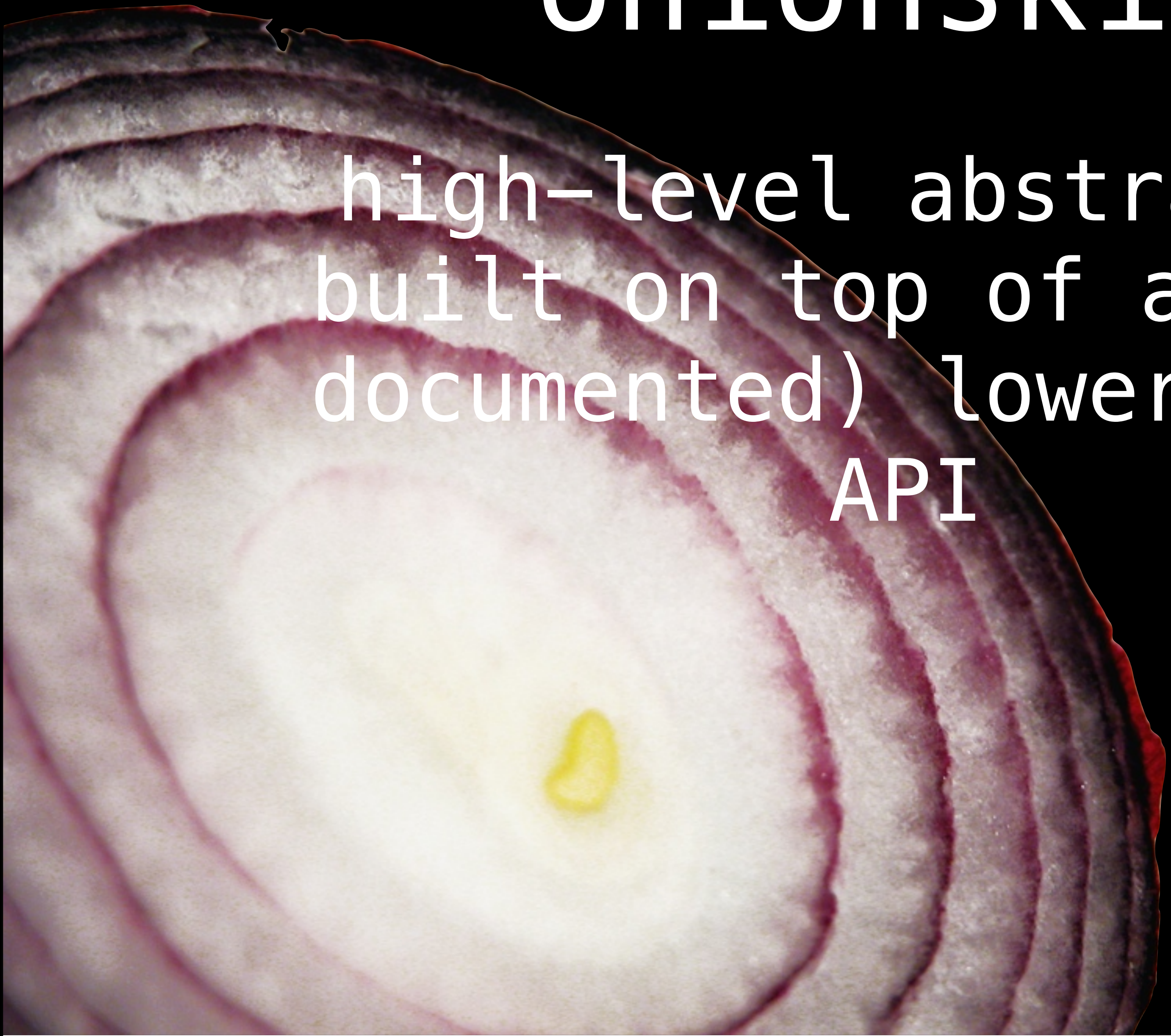


"some leak in better ways than others"

"the best abstractions leak intentionally, in carefully chosen ways"

Glenn Vanderburg

onionskin API



high-level abstraction built on top of a (well documented) lower-level API

Rails ActiveRecord

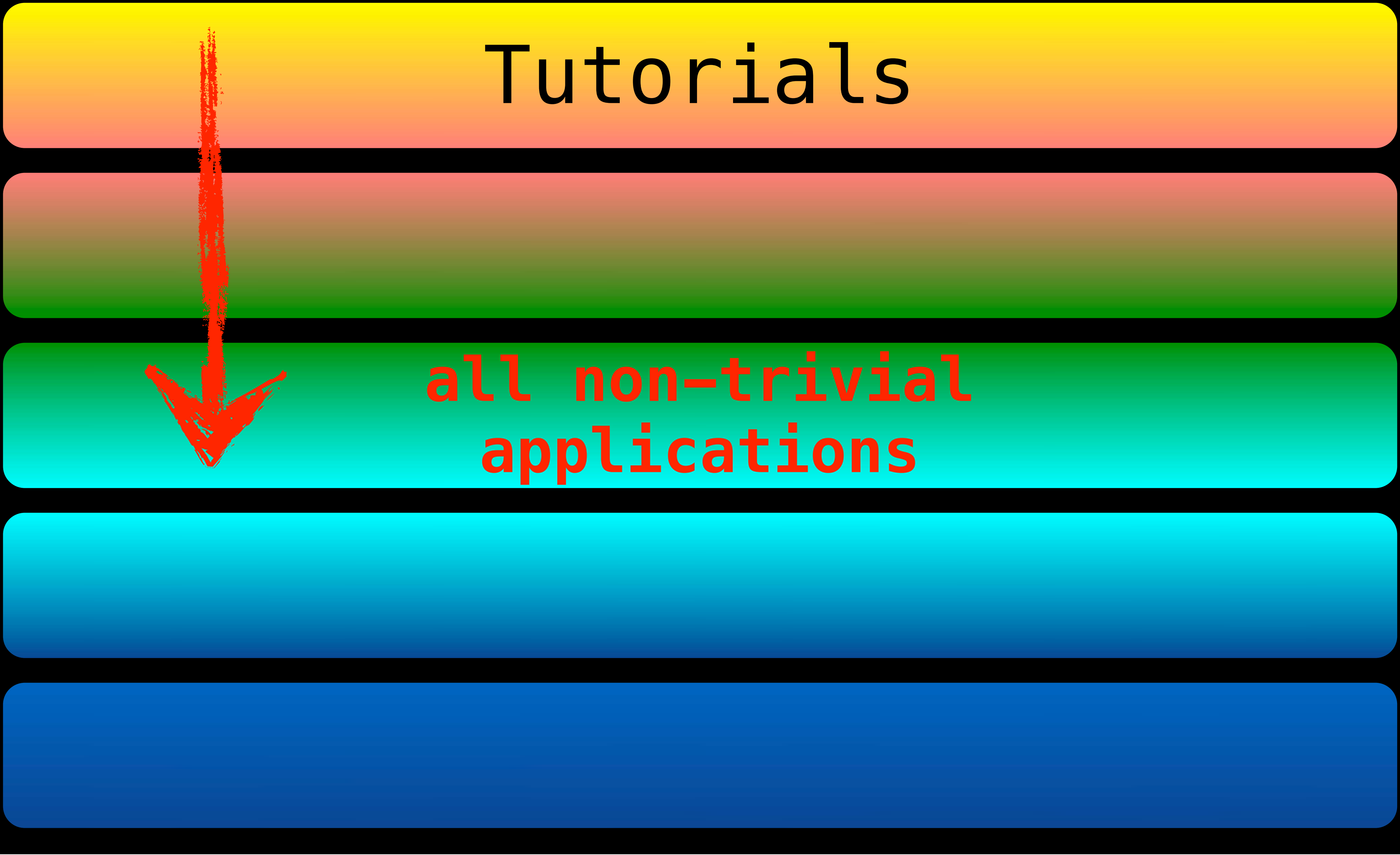
query instances of particular classes with associations

queries as high-level relational algebra methods

SQL fragments composed into SELECT statements

query =>object instances

vendor agnostic database connections and facilities

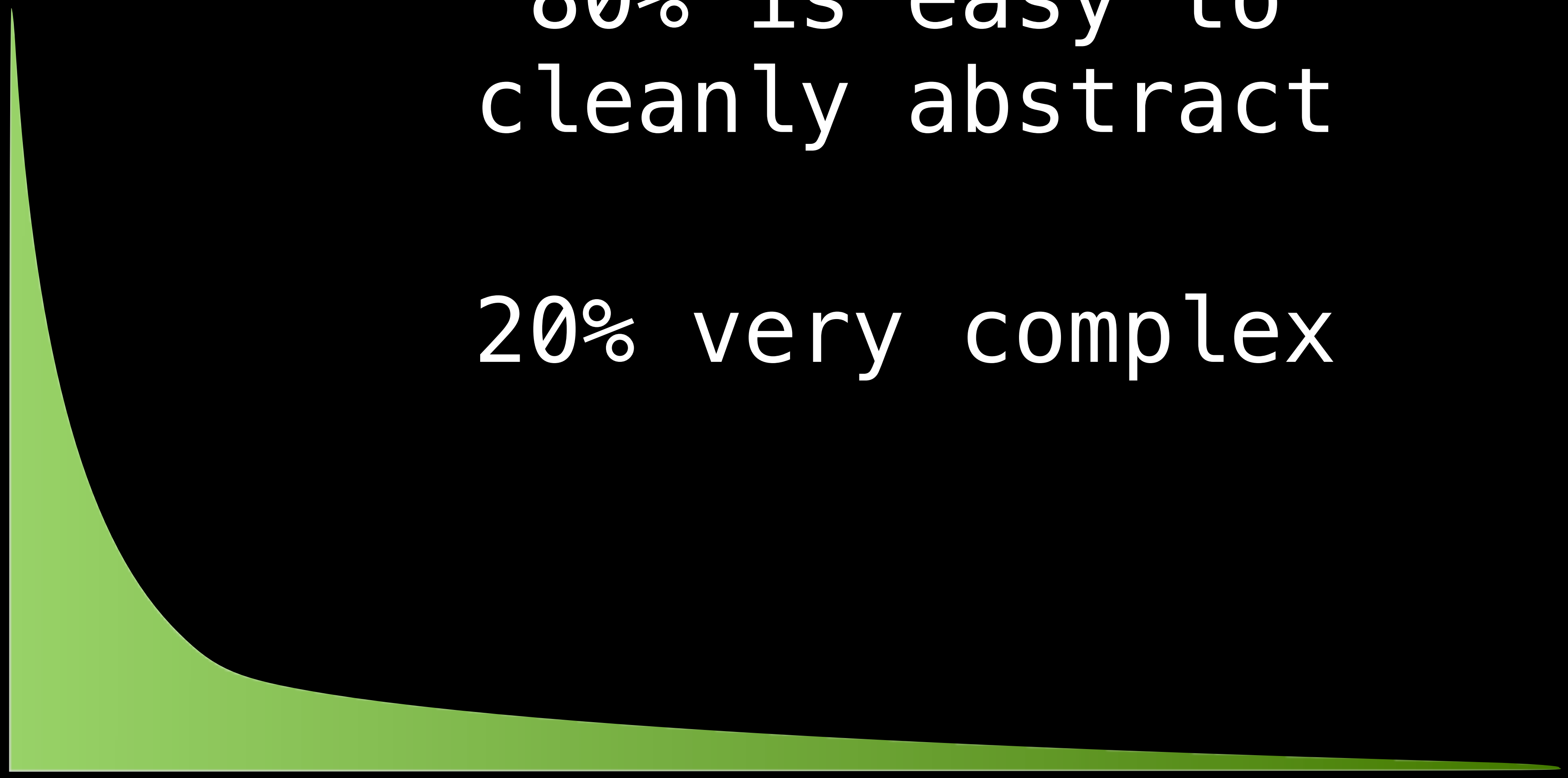


Tutorials

all non-trivial applications

Lesson #7

Good APIs are not merely high-level or low-level; they're both at once.



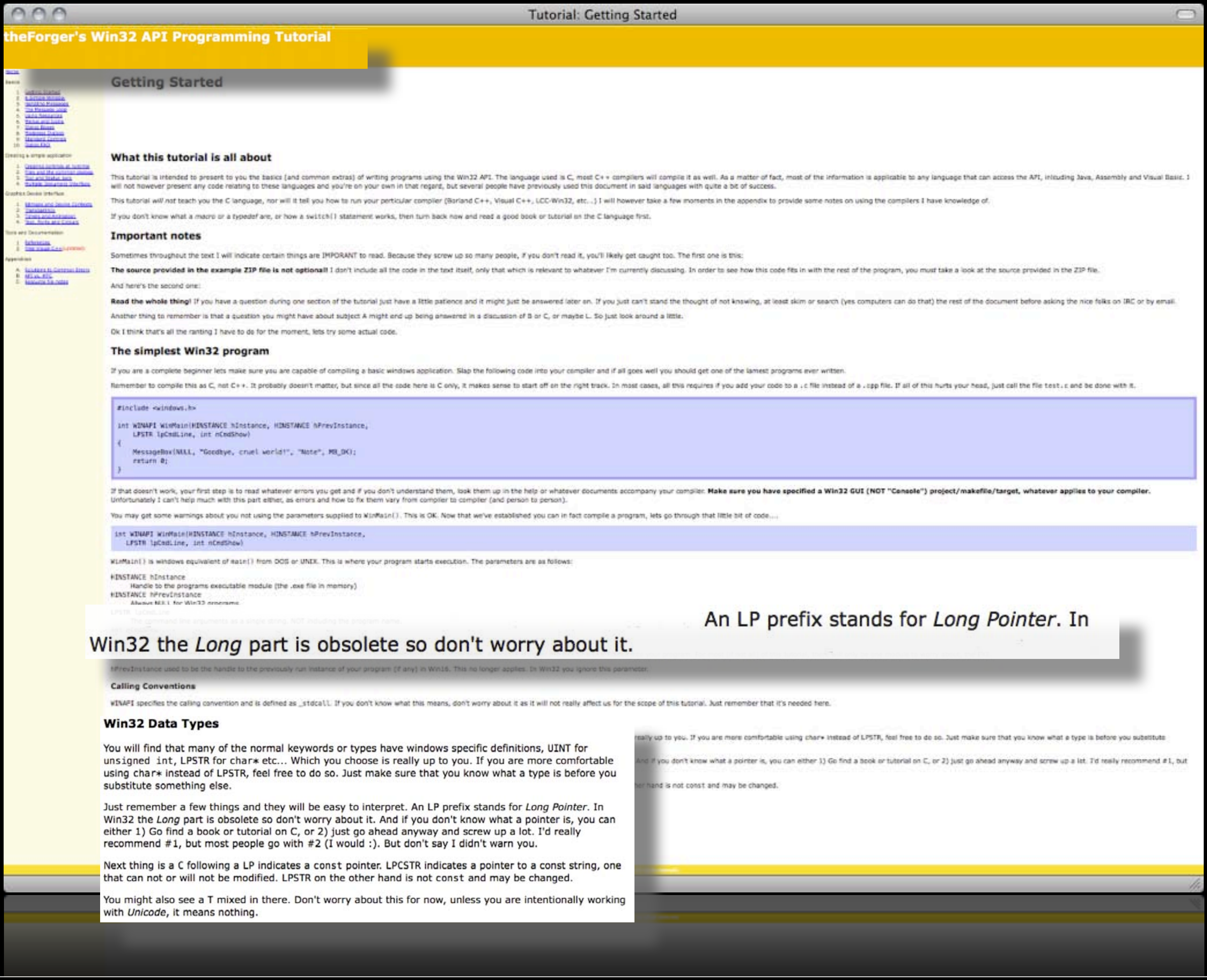
80% is easy to cleanly abstract

20% very complex

Lesson #8:

Generalize the 80% cases; get out of the way for the rest.

lpstr



theForger's Win32 API Programming Tutorial

Getting Started

What this tutorial is all about

This tutorial is intended to present to you the basics (and common extras) of writing programs using the Win32 API. The language used is C, most C++ compilers will compile it as well. As a matter of fact, most of the information is applicable to any language that can access the API, inlcuding Java, Assembly and Visual Basic. I will not however present any code relating to these languages and you're on your own in that regard, but several people have previously used this document in said languages with quite a bit of success.

This tutorial *will not* teach you the C language, nor will it tell you how to run your perticular compiler (Borland C++, Visual C++, LCC-Win32, etc...) I will however take a few moments in the appendix to provide some notes on using the compilers I have knowledge of.

If you don't know what a *macro* or a *typedef* are, or how a `switch()` statement works, then turn back now and read a good book or tutorial on the C language first.

Important notes

Sometimes throughout the text I will indicate certain things are IMPORANT to read. Because they screw up so many people, *if you don't read it, you'll likely get caught too.* The first one is this:

The source provided in the example ZIP file is not optional! I don't include all the code in the text itself, only that which is relevant to whatever I'm currently discussing. In order to see how this code fits in with the rest of the program, you must take a look at the source provided in the ZIP file.

And here's the second one:

Read the whole thing! If you have a question during one section of the tutorial just have a little patience and it might just be answered later on. If you just can't stand the thought of not knowing, at least skim or search (yes computers can do that) the rest of the document before asking the nice folks on IRC or by email.

Another thing to remember is that a question you might have about subject A might end up being answered in a discussion of B or C, or maybe L. So just look around a little.

Ok I think that's all the ranting I have to do for the moment, lets try some actual code.

The simplest Win32 program

If you are a complete beginner lets make sure you are capable of compiling a basic windows application. Slap the following code into your compiler and if all goes well you should get one of the lamest programs ever written.

Remember to compile this as C, not C++. It probably doesn't matter, but since all the code here is C only, it makes sense to start off on the right track. In most cases, all this requires if you add your code to a `.c` file instead of a `.cpp` file. If all of this hurts your head, just call the file `test.c` and be done with it.

```
#include <windows.h>

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
    LPSTR lpCmdLine, int nCmdShow)
{
    MessageBox(NULL, "Goodbye, cruel world!", "Note", MB_OK);
    return 0;
}
```

If that doesn't work, your first step is to read whatever errors you get and if you don't understand them, look them up in the help or whatever documents accompany your compiler. **Make sure you have specified a Win32 GUI (NOT "Console") project/makefile/target, whatever applies to your compiler.** Unfortunately I can't help much with this part either, as errors and how to fix them vary from compiler to compiler (and person to person).

You may get some warnings about you not using the parameters supplied to `WinMain()`. This is OK. Now that we've established you can in fact compile a program, lets go through that little bit of code....

```
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
   LPSTR lpCmdLine, int nCmdShow)
```

`WinMain()` is windows equivalent of `main()` from DOS or UNIX. This is where your program starts execution. The parameters are as follows:

`HINSTANCE hInstance`
    Handle to the programs executable module (the .exe file in memory)
`HINSTANCE hPrevInstance`
    Always NULL for Win32 programs
`LPSTR lpCmdLine`
    The command line arguments as a single string. NOT including the program name.
`int nCmdShow`

An LP prefix stands for *Long Pointer*. In Win32 the *Long* part is obsolete so don't worry about it.

`hPrevInstance` used to be the handle to the previously run instance of your program (if any) in Win16. This no longer applies. In Win32 you ignore this parameter.

Calling Conventions

`WINAPI` specifies the calling convention and is defined as `_stdcall`. If you don't know what this means, don't worry about it as it will not really affect us for the scope of this tutorial. Just remember that it's needed here.

Win32 Data Types

You will find that many of the normal keywords or types have windows specific definitions, UINT for `unsigned int`, LPSTR for `char*` etc... Which you choose is really up to you. If you are more comfortable using `char*` instead of LPSTR, feel free to do so. Just make sure that you know what a type is before you substitute something else.

Just remember a few things and they will be easy to interpret. An LP prefix stands for *Long Pointer*. In Win32 the *Long* part is obsolete so don't worry about it. And if you don't know what a pointer is, you can either 1) Go find a book or tutorial on C, or 2) just go ahead anyway and screw up a lot. I'd really recommend #1, but most people go with #2 (I would :). But don't say I didn't warn you.

Next thing is a C following a LP indicates a `const` pointer. LPCSTR indicates a pointer to a const string, one that can not or will not be modified. LPSTR on the other hand is not `const` and may be changed.

You might also see a T mixed in there. Don't worry about this for now, unless you are intentionally working with *Unicode*, it means nothing.

Tutorial: Getting Started
theForger's Win32 API Programming Tutorial

Getting Started

What this tutorial is all about

Important notes

The simplest Win32 program

Win32 Data Types

You will find that many of the normal keywords or types have windows specific definitions, UINT for unsigned int, LPSTR for char* etc... Which you choose is really up to you. If you are more comfortable using char* instead of LPSTR, feel free to do so. Just make sure that you know what a type is before you substitute something else.

Just remember a few things and they will be easy to interpret. An LP prefix stands for *Long Pointer*. In Win32 the *Long* part is obsolete so don't worry about it. And if you don't know what a pointer is, you can either 1) Go find a book or tutorial on C, or 2) just go ahead anyway and screw up a lot. I'd really recommend #1, but most people go with #2 (I would :). But don't say I didn't warn you.

Next thing is a C following a LP indicates a const pointer. LPCSTR indicates a pointer to a const string, one that can not or will not be modified. LPSTR on the other hand is not const and may be changed.

You might also see a T mixed in there. Don't worry about this for now, unless you are intentionally working with *Unicode*, it means nothing.

abstraction

distraction

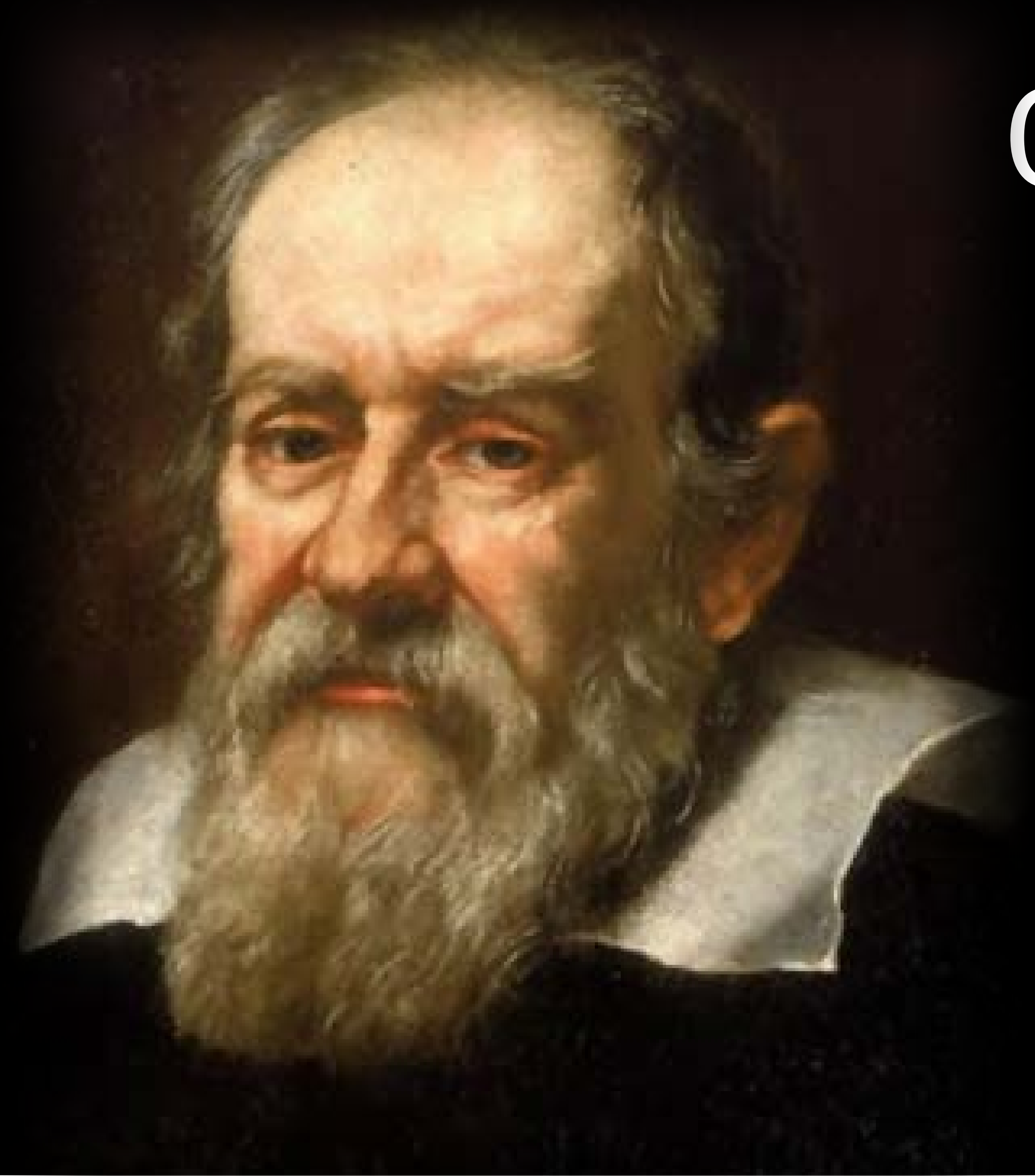
Lesson #4:

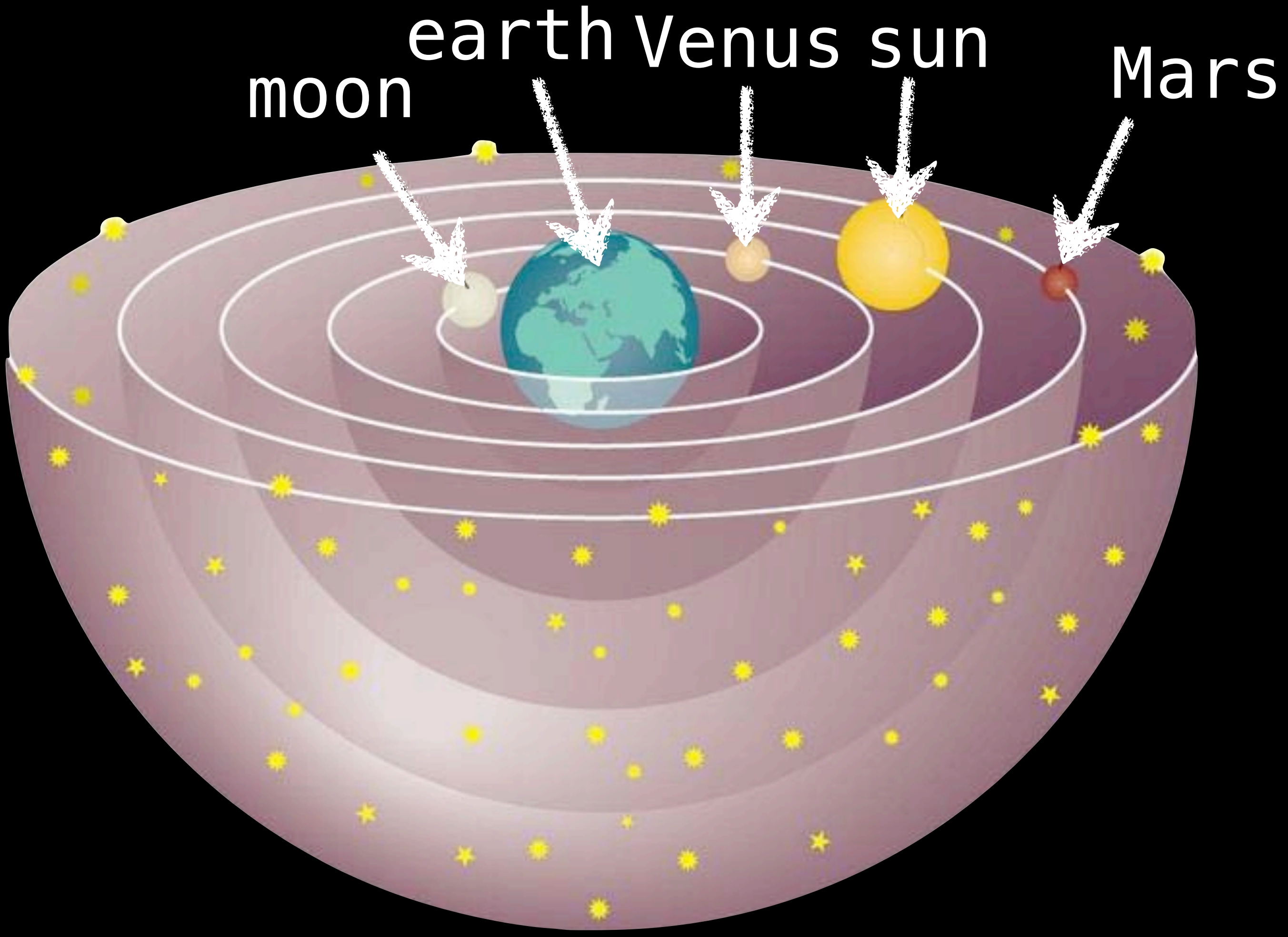
Don't name things that expose underlying details.

Lesson #6:

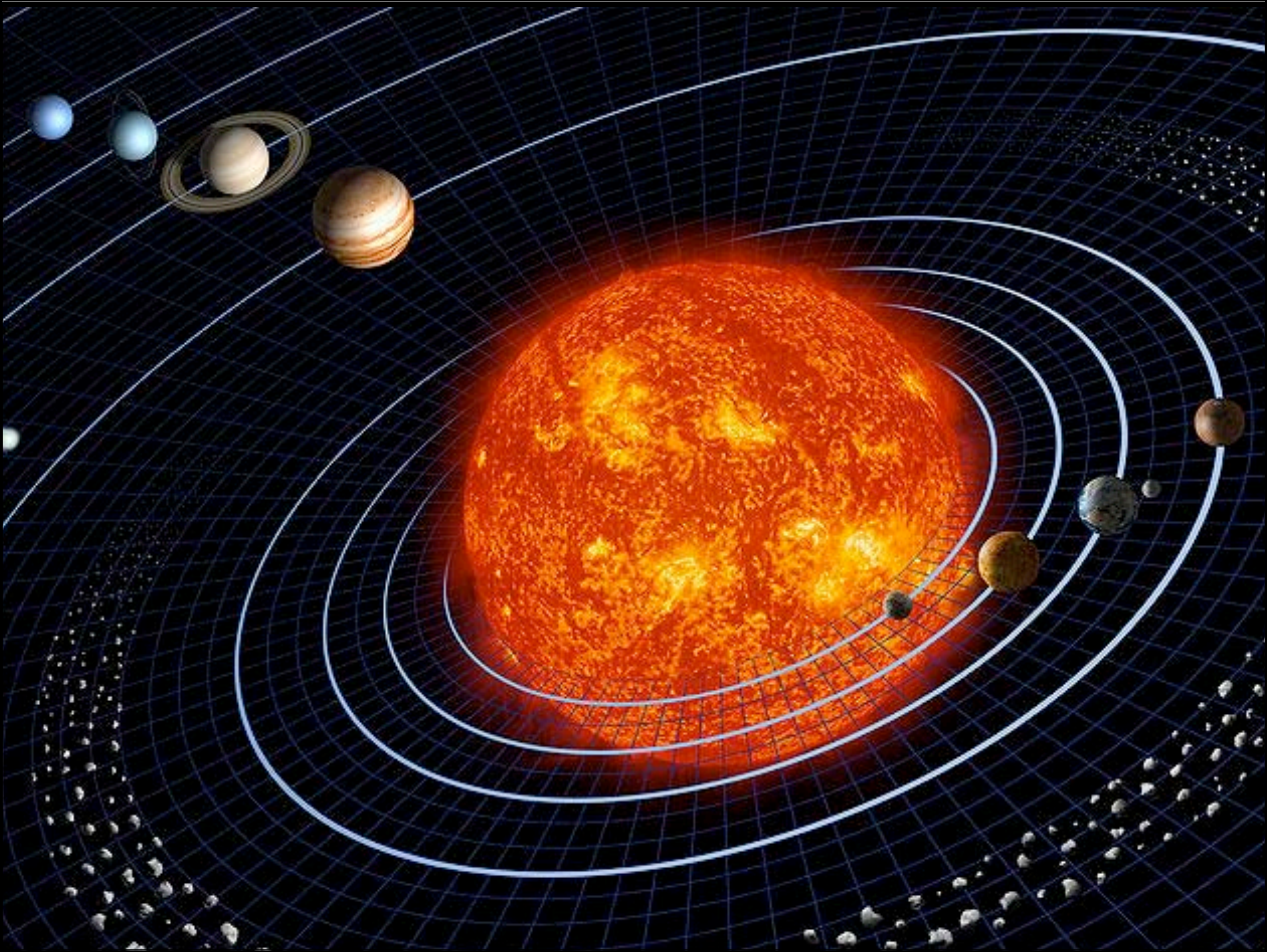
Understand the implications of rigidity

Galileo





moon
earth
Venus
sun
Mars



Maven

http://kent.spillner.org/blog/work/2009/11/14/java-build-tools.html

“Maven builds are an infinite cycle of despair that will slowly drag you into the deepest, darkest pits of hell (where Maven itself was forged).”

composable

BASH

Rake Gant

Emacs

languages

contextual

PowerShell

<APACHE ANT>

maven

eclipse

frameworks

context isn’t all bad

> context

> “out of the box”

> contextual intelligence

< flexibility

composable

< implicit behavior

> building blocks

> eventual power

< initial power

> flexible

Dietzler’s Law

for tool ***X***:

80% of what user wants → fast & easy

the next 10% → possible but difficult

the last 10% → *impossible*

users want 100% of what they want

choosing

always start with easiest

at some point, you must switch to something more powerful

you can’t have both

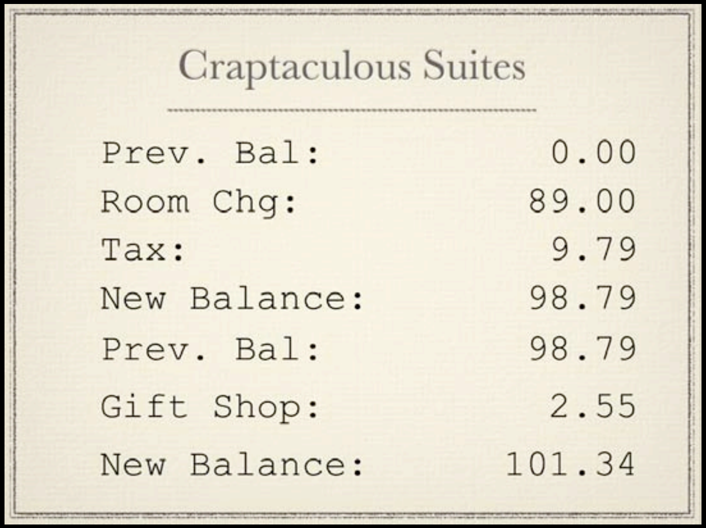
how do you replace something seemingly useful?

this is a hard decision!

maven

the 1 true abstraction?

composability



Craptaculous Suites

Prev. Bal:	0.00
Room Chg:	89.00
Tax:	9.79
New Balance:	98.79
Prev. Bal:	98.79
Gift Shop:	2.55
New Balance:	101.34

≠

Lesson #9:

Understand 1 level below your usual abstraction.

understand your abstractions

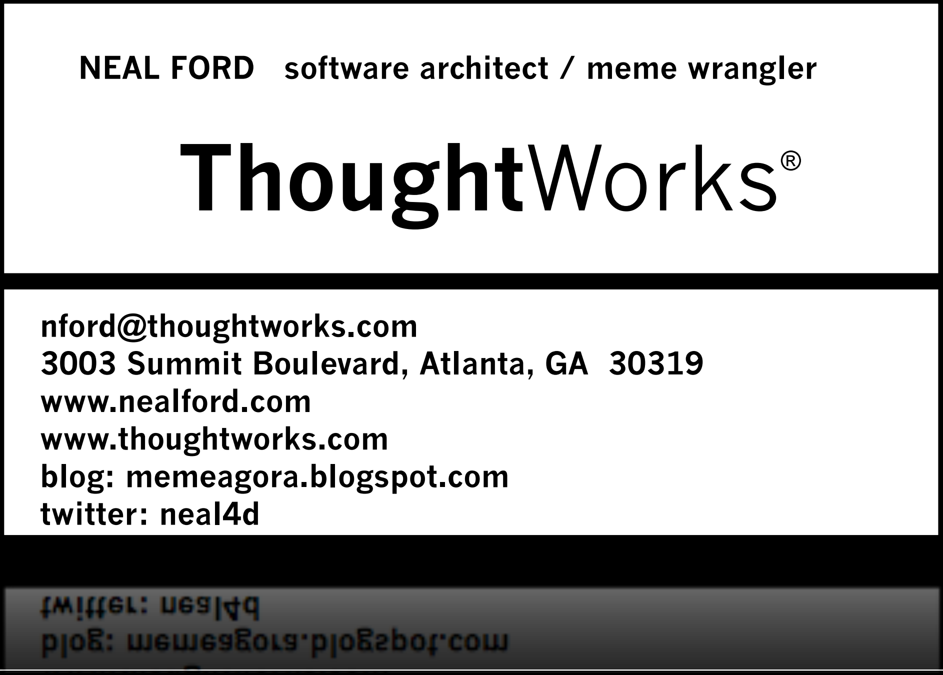
understand 1 level
below your
abstractions

don’t hate…
learn & understand

don’t be distracted
by your abstractions

go home

thanks for coming



NEAL FORD software architect / meme wrangler

ThoughtWorks®

nford@thoughtworks.com
3003 Summit Boulevard, Atlanta, GA 30319
www.nealford.com
www.thoughtworks.com
blog: memeagora.blogspot.com
twitter: neal4d