



# The Lean Tech Stack

## Engineering Nimble Experiences

Bill Scott

Sr. Director UI Engineering

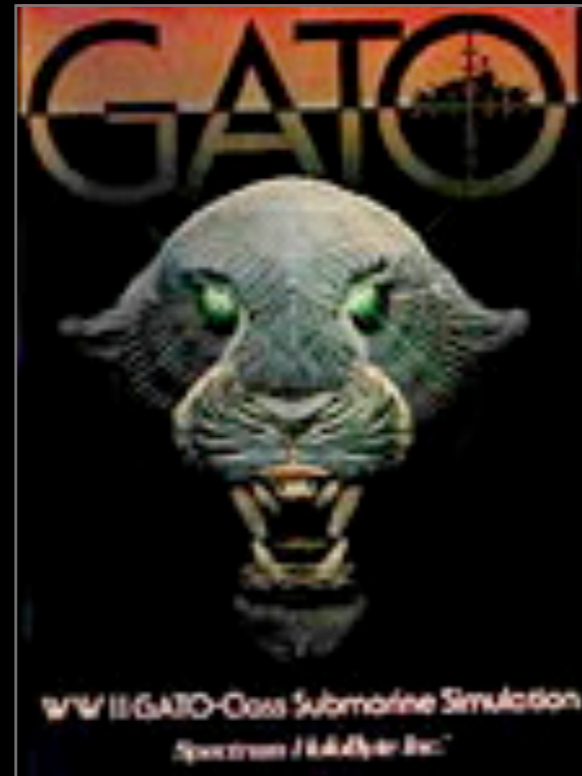
PayPal

July 14

Open Web Camp 2012



# The Way it Was

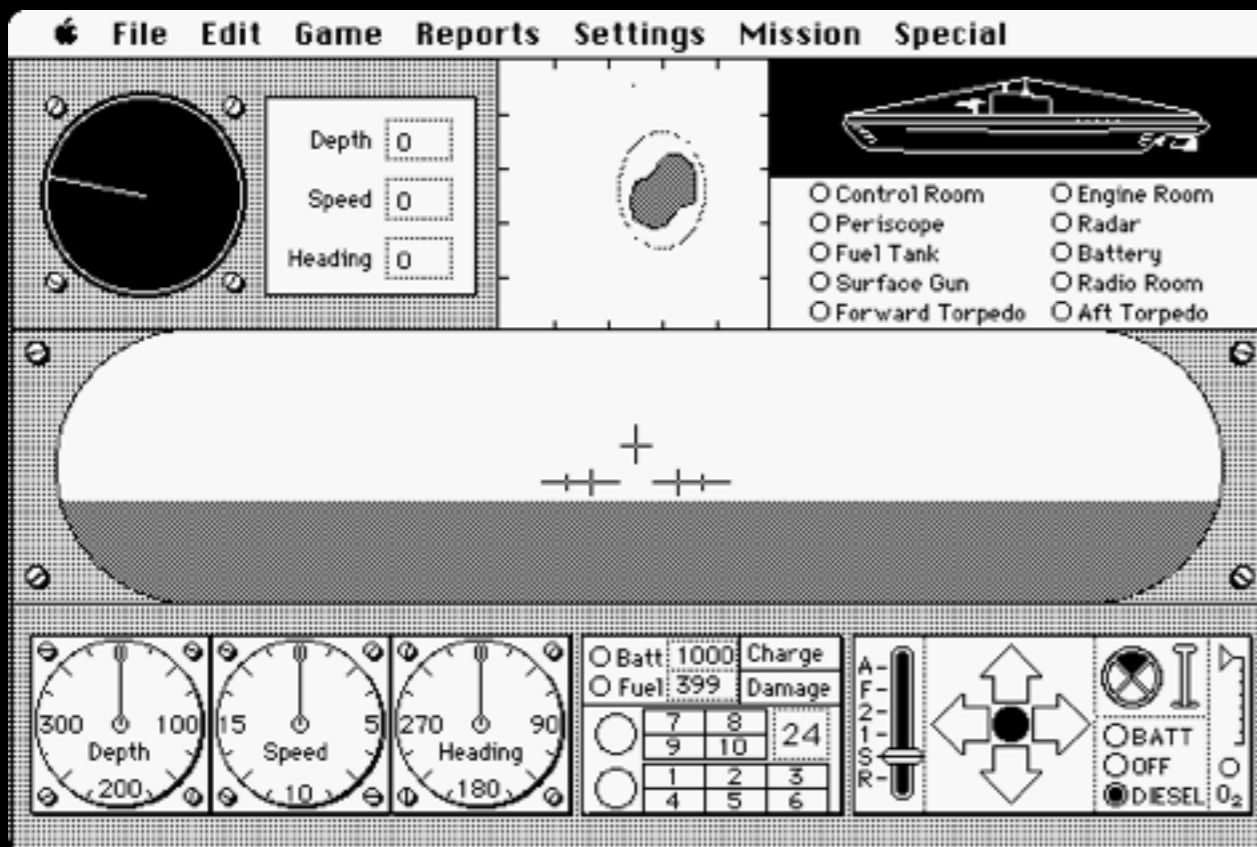


# Building a game in 1985

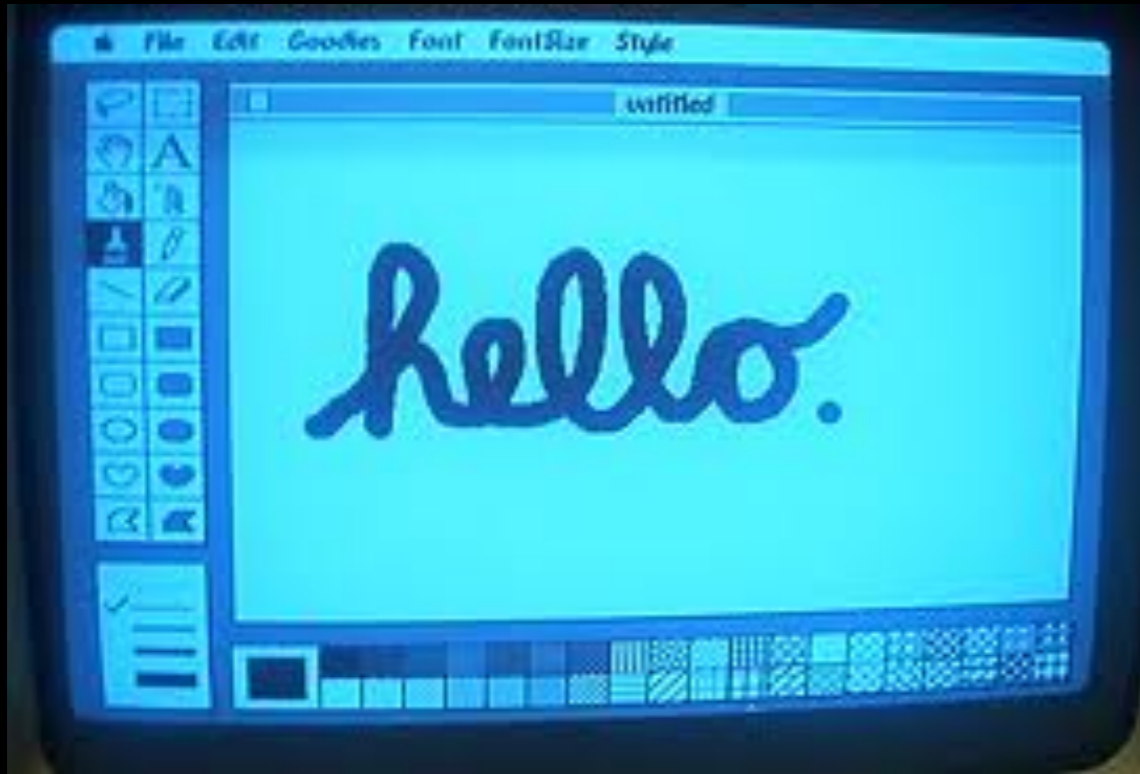
Original Mac Quickdraw Toolkit provided some GUI framework pieces (like the Open File Box)

This bit of “path of least resistance” was a powerful boost to consistency and creating nice looking Macintosh UIs

But there was still a lot left to create on your own



# Developing a UI was still hard work



No internet. Open source was practically non-existent.

Hard to build (all native & assembly)

Long shelf life (long release cycles with 3.5" disk deployment)

Reuse was something we longed for

# Developing a UI was still hard work



No internet. Open source was practically non-existent.

Hard to build (all native & assembly)

Long shelf life (long release cycles with 3.5" disk deployment)

Reuse was something we longed for

# Developing a UI was still hard work



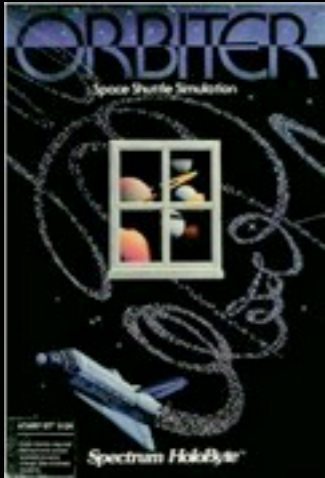
No internet. Open source was practically non-existent.

Hard to build (all native & assembly)

Long shelf life (long release cycles with 3.5" disk deployment)

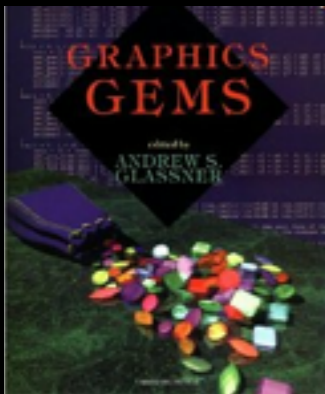
Reuse was something we longed for

# '85-'05. Lots of proprietary frameworks.



Orbiter. 3D Graphics Library

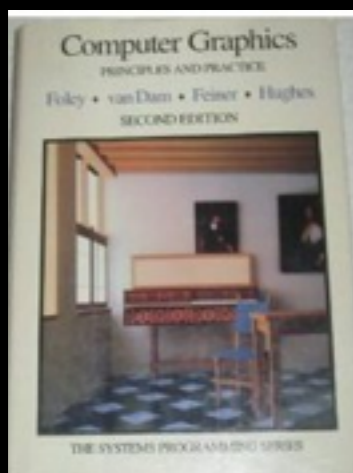
ESYView. Wargame simulator & briefing tool. Almost everything had to be written from the ground up (mapping projection library, double buffering, display lists, canvas)



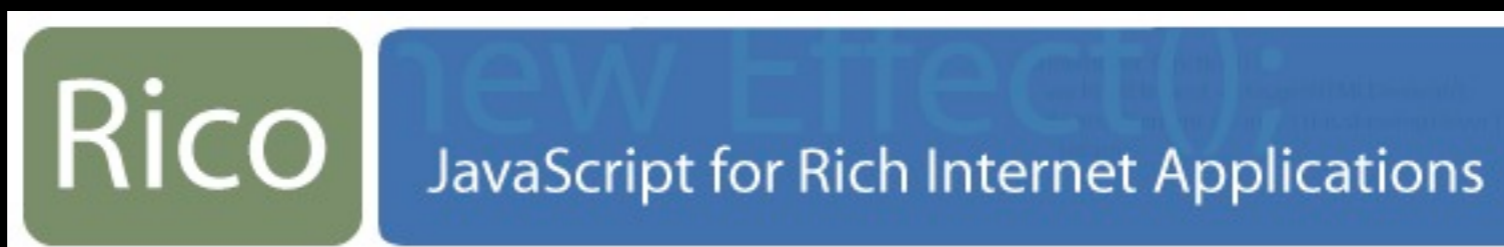
C++ Frameworks for Motif

Tcl/TK Frameworks

3 JSP Frameworks (i2 Technologies, Sabre, Netflix)



Rico. One of first Ajax/JS frameworks 2005.



# Sharing with the world



Rico. One of the early Ajax/JS frameworks (2005)



Launched Yahoo! Design Pattern Library (2006)



Worked closely with YUI team (and built first carousel version)

# Sharing with the world



Rico. One of the early Ajax/JS frameworks (2005)



Launched Yahoo! Design Pattern Library (2006)



Worked closely with YUI team (and built first carousel version)

Lesson: The power of sharing and multiplying!



# The Epiphany

# Netflix way of working



Different way of working, different culture

# Netflix way of working



Different way of working, different culture

## 1. Working Model - Experience-Driven

Start with experience

Only customer is the member (no internal customers)

We rarely talked about the “machine”. We talked about members.

# Netflix way of working



Different way of working, different culture

## 1. Working Model - Experience-Driven

Start with experience

Only customer is the member (no internal customers)

We rarely talked about the “machine”. We talked about members.

## 2. Culture - Rapid Experimentation

Get it out live as fast as possible

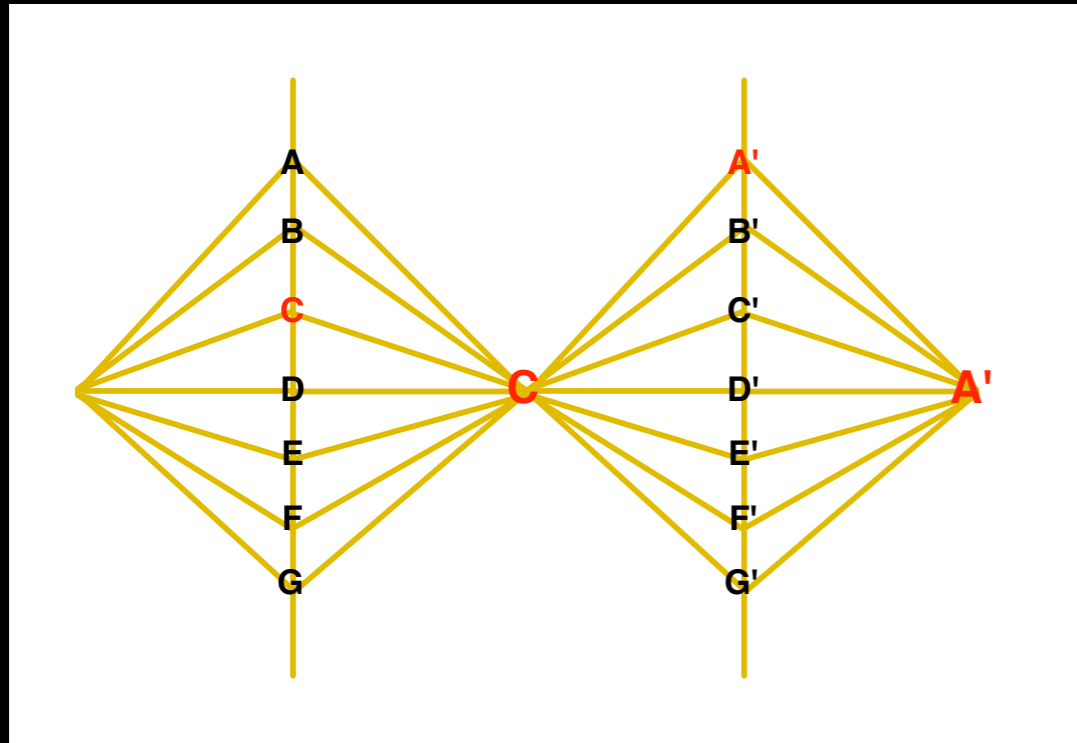
Fail fast

Learn fast

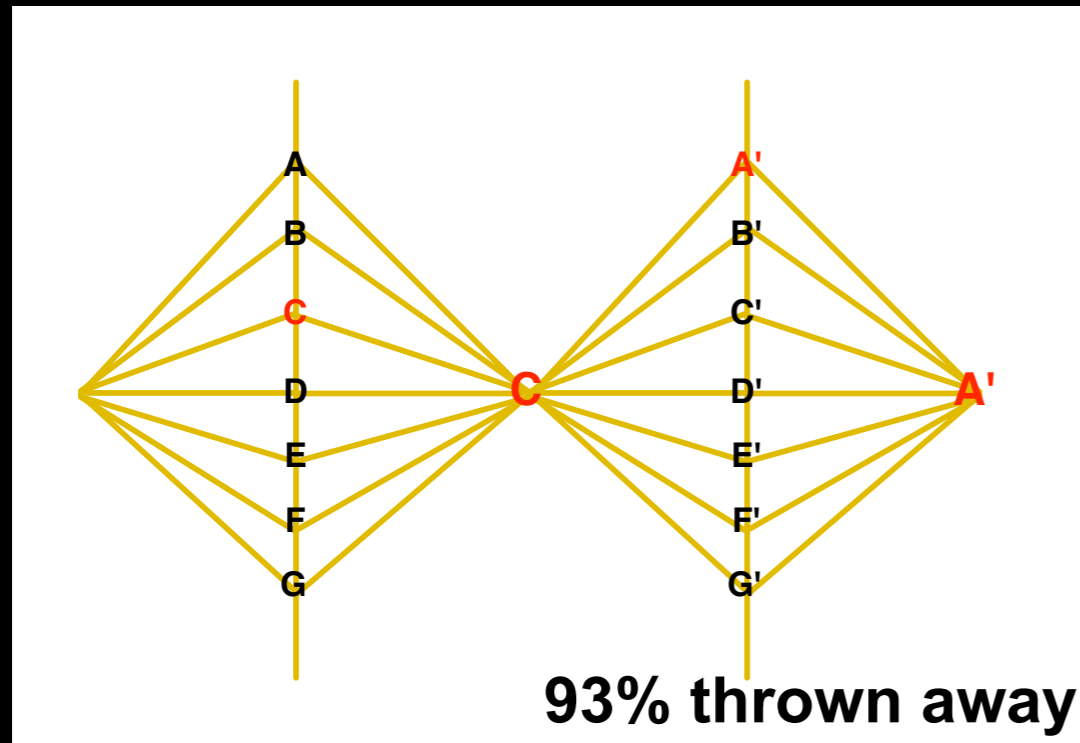
Don't over think it

# Epiphany

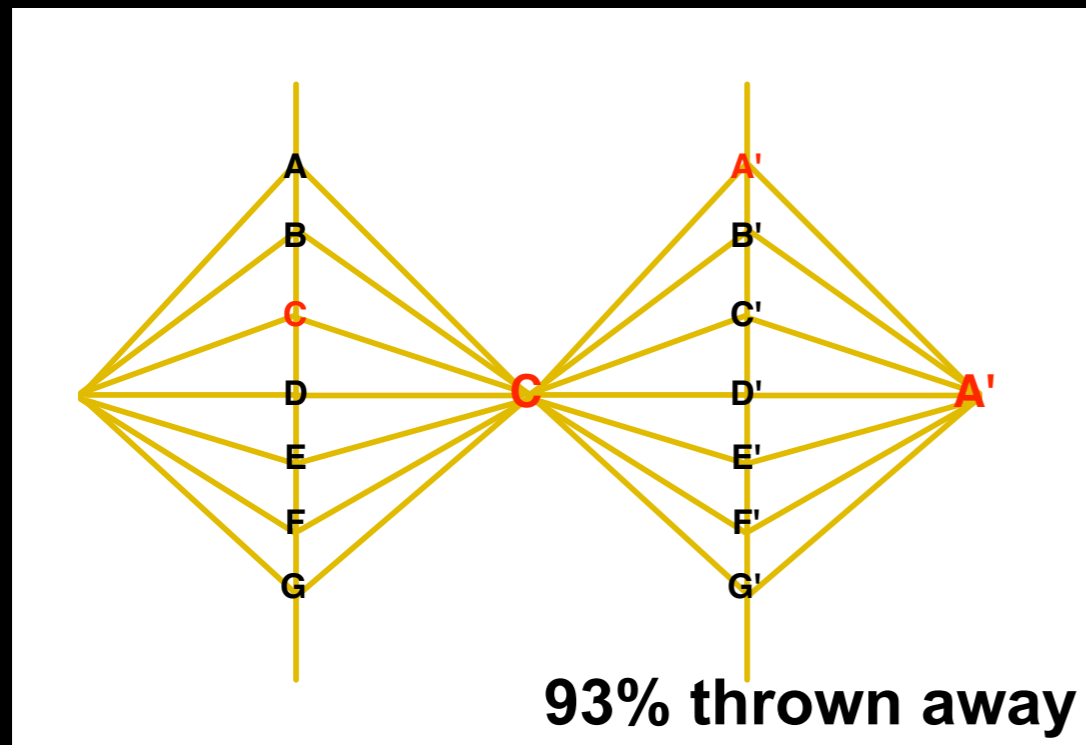
# Epiphany



# Epiphany



# Epiphany

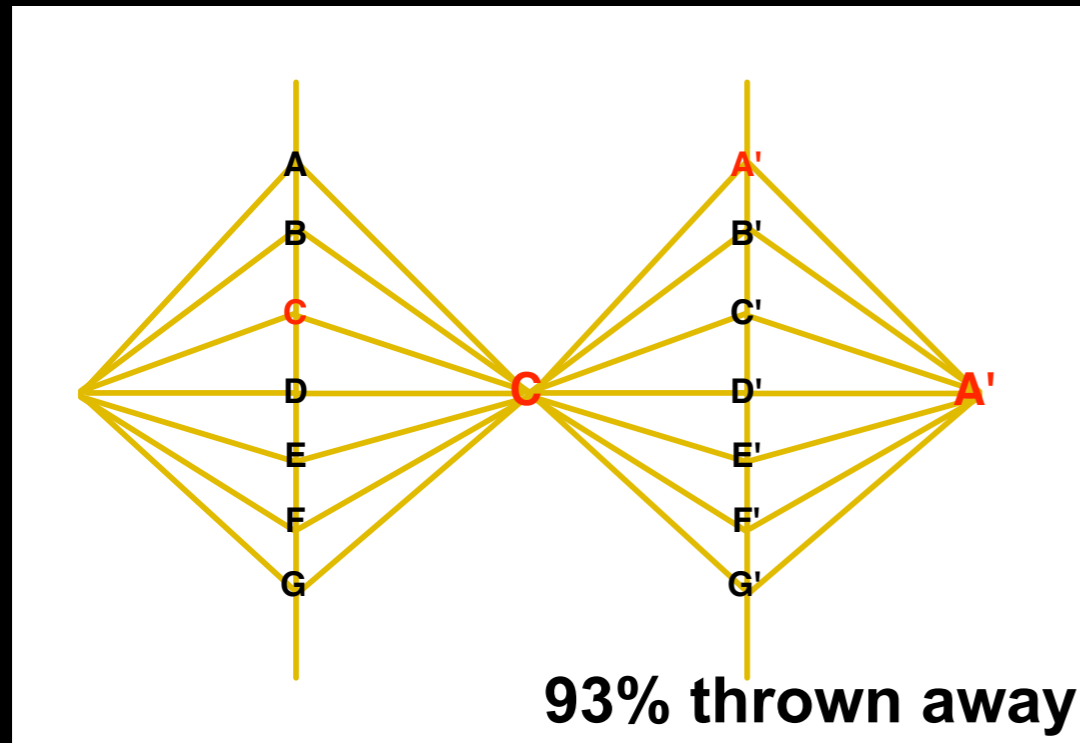


Follow Build-Test-Learn  
Design for volatility, throwaway-ability

At Netflix 90% or more of the UI code was thrown away every year

Doesn't take too many tests to result in lots of throw away code

# Epiphany



Follow Build-Test-Learn  
Design for volatility, throwaway-ability

At Netflix 90% or more of the UI code was thrown away every year

Doesn't take too many tests to result in lots of throw away code

**UI Layer =  
Experimentation Layer**

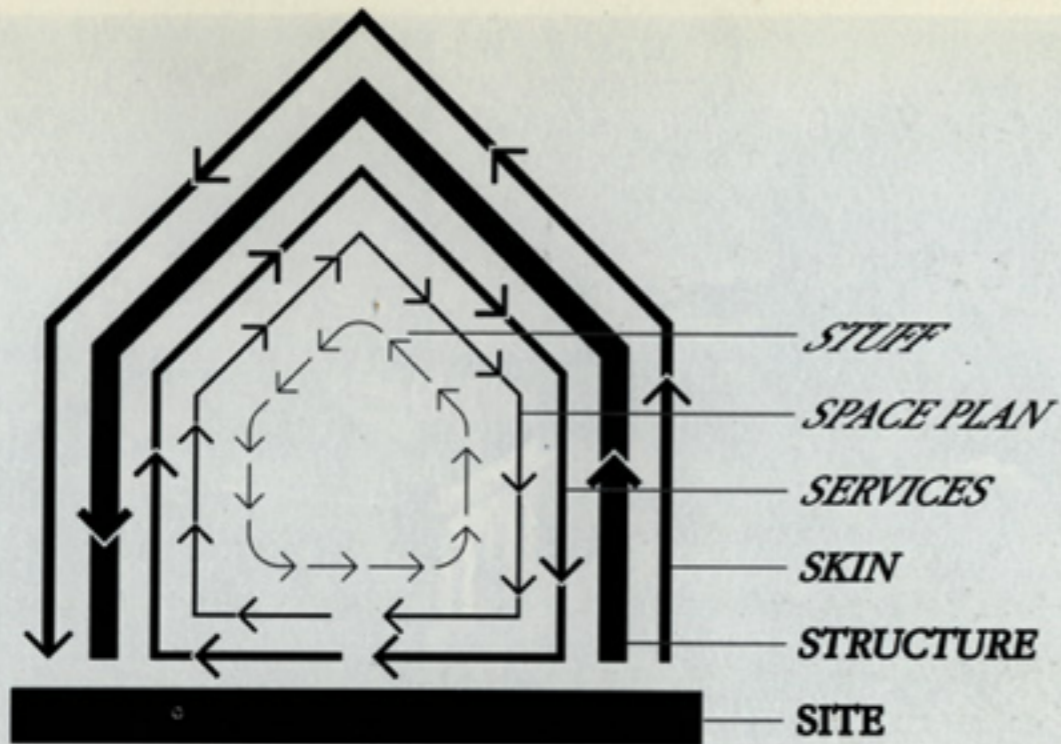
A droid, resembling a Star Wars character, stands in the lower right foreground, facing away from the viewer and looking up at a wall. The wall is covered in a repeating pattern of the phrase "I WILL FIND THE DROIDS I'M LOOKING FOR" written in a light, chalk-like substance. A black rectangular box with a white border is superimposed over the middle of the image, containing the text "Lessons Learned" in white. In the bottom left corner, the text "Engineering experiences in a nimble world" is written in white.

# Lessons Learned

Engineering experiences  
in a nimble world

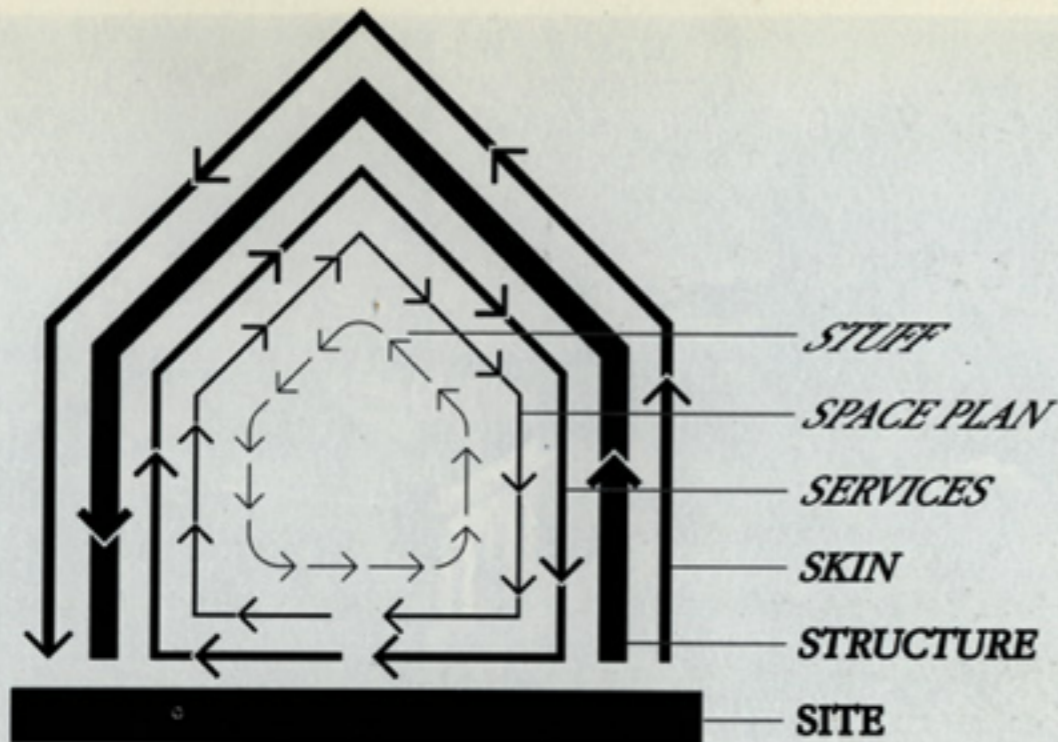


# **#1 Design for Volatility**



Donald Ryan

**SHEARING LAYERS OF CHANGE.** Because of the different rates of change of its components, a building is always tearing itself apart.



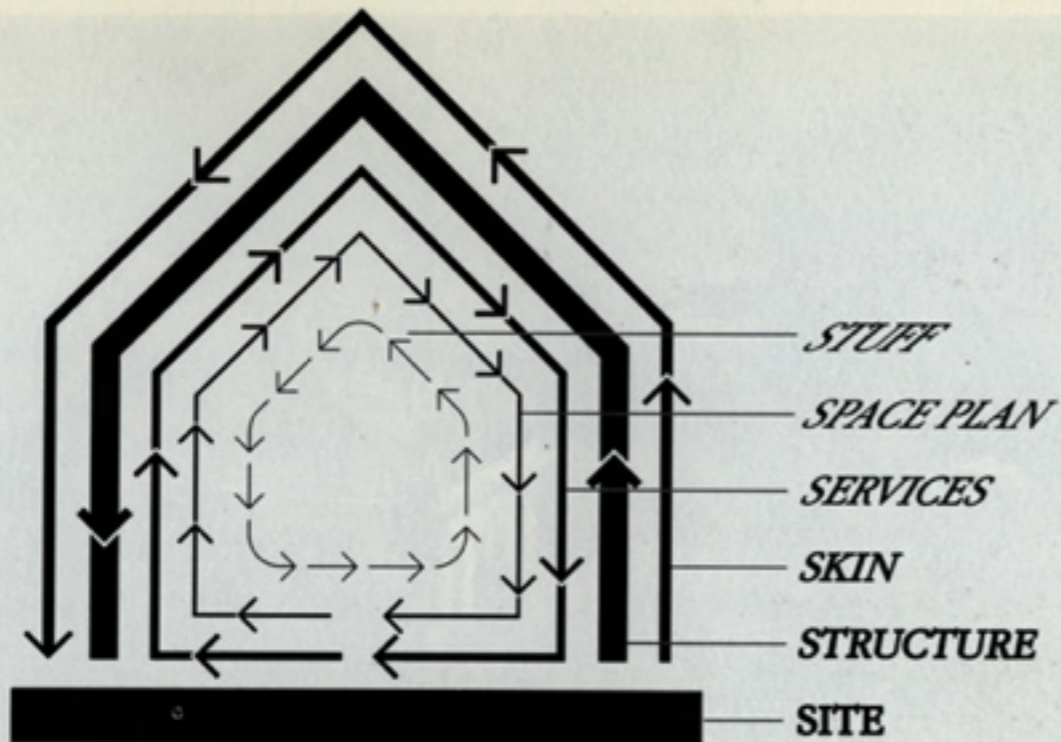
Donald Ryan

**SHEARING LAYERS OF CHANGE.** Because of the different rates of change of its components, a building is always tearing itself apart.

## User Interface Shears Fastest

Our software is always tearing itself apart (or should be)

Recognize that different layers change at different velocities



Donald Ryan

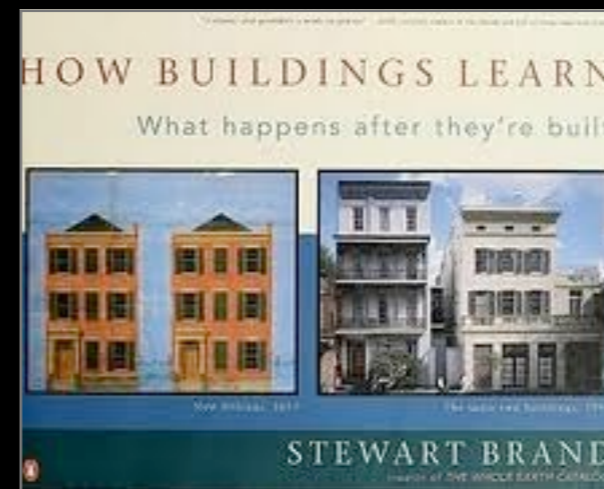
**SHEARING LAYERS OF CHANGE.** Because of the different rates of change of its components, a building is always tearing itself apart.

*Stewart Brand - How Buildings Learn*

# User Interface Shears Fastest

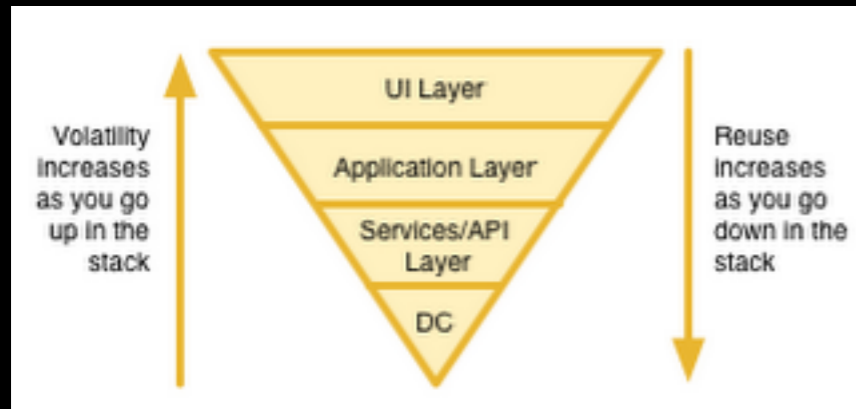
Our software is always tearing itself apart (or should be)

Recognize that different layers change at different velocities

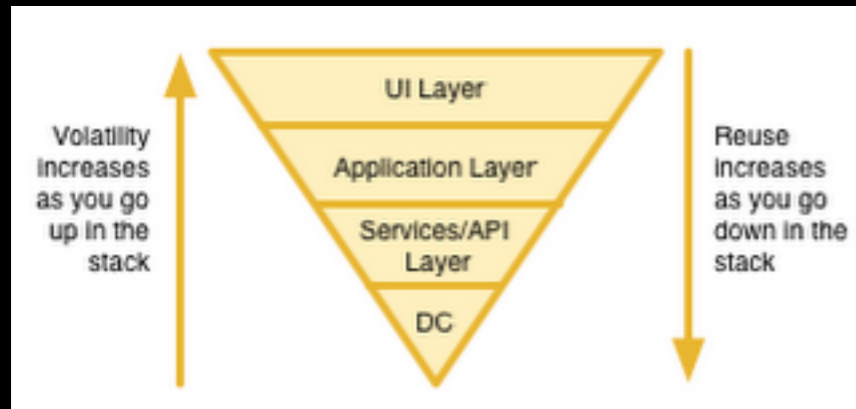


*All buildings are predictions. All predictions are wrong. There's no escape from this grim syllogism, but it can be softened. - Brand*

# Design at different velocities by layer



# Design at different velocities by layer



Recognize that different parts of Stack change at different velocities

"any building is actually a hierarchy of pieces, each of which inherently changes at different rates"

- Stewart Brand. How Buildings Learn.

Design for throwaway-ability (Volatility)!

Use before you Reuse (optimize for change)

Utilize packaging or Paths to capture experiments

A close-up photograph of a paint palette. The palette is covered with thick, textured paint in various colors. On the left, there are large dollops of bright red and orange paint. In the center and right, there are smaller amounts of red, yellow, and blue paint. The background is dark and out of focus. A black text box with a white border is overlaid on the right side of the image, containing the text "#2 Start with the Experience" in white, bold, sans-serif font.

**#2 Start with  
the Experience**

# Experience vs Components

# Experience vs Components

NETFLIX

2 / 332

Search

## Instant Queue



## Bella

2006 PG-13 1h 31m



Two lost souls – Nina, a pregnant, unmarried waitress, and Jose, an introspective cook with a tragic past – find solace in each other as their lives become unpredictably linked throughout the course of one incredible day.

Cast: Eduardo Verástegui, Tammy Blanchard...

Categories: Drama, Indie Dramas

Director: Alejandro Gomez Monteverde

## Recently Watched



## Emotional Dramas

Watch Instantly

Browse DVDs

Your Queue

★ Suggestions For You

Movies, TV shows, actors, directors, genres 🔍

Genres ▾

New Arrivals

Starz Play

Instantly to your TV

## You recently watched:

[See all](#)

Danny Phantom: Ssn 2: Reality Tr ...



Play Next

Heroes: Ssn 1: Homecoming



Resume

2m 43m

Alice in Wonderland



Resume

0m 108m

Bill, rate what you've seen to reveal suggestions just for you

Rate *Heroes:*  
*Season 1*

Haven't Seen It

Suggestion

1

Suggestion

2

Suggestion

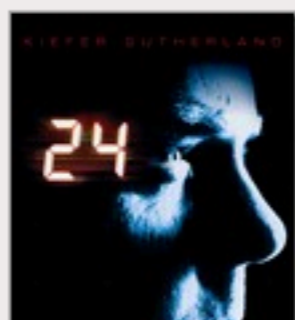
3

## Suspenseful Conspiracy Action &amp; Adventure

[See all >](#)Your taste preferences  
created this row.Suspenseful  
Action & Adventure

As well as your interest in...

24: Season 2



Chain Reaction



Westbound



Boxer's Adventure



Watch Instantly

Browse DVDs

Your Queue

★ Suggestions For You

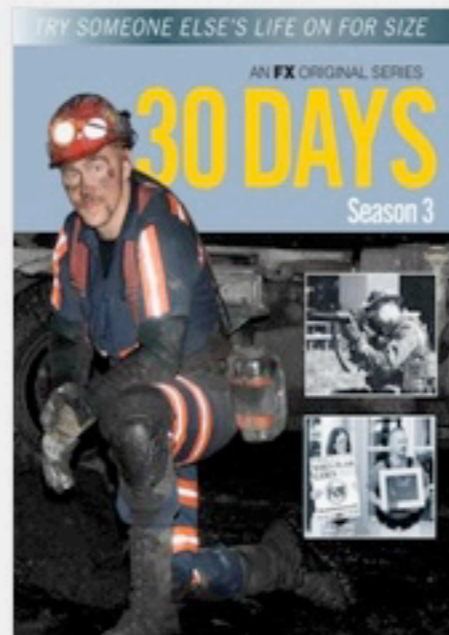
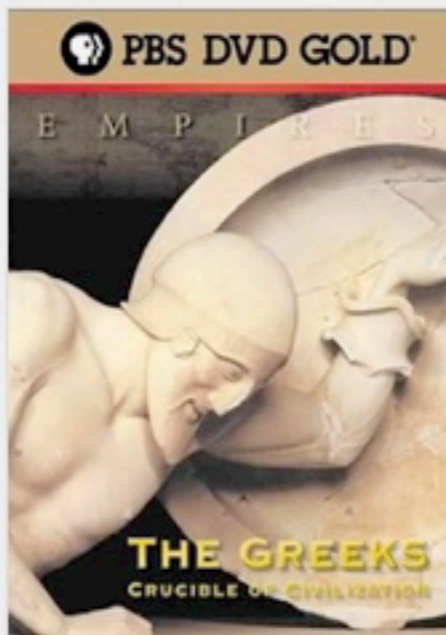
Genres ▼

New Arrivals

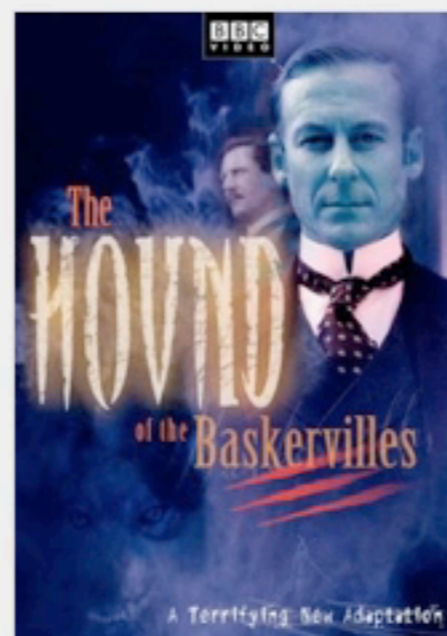
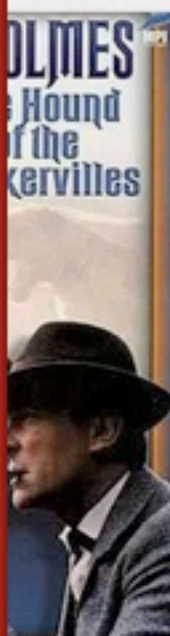
Starz Play

Instantly to your TV

## TOP 10 FOR BILL



## BRITISH CRIME THRILLERS



## CRITICALLY-ACCLAIMED DARK ACTION & ADVENTURE

Watch Instantly Browse DVDs Your Queue ★ Suggestions For You

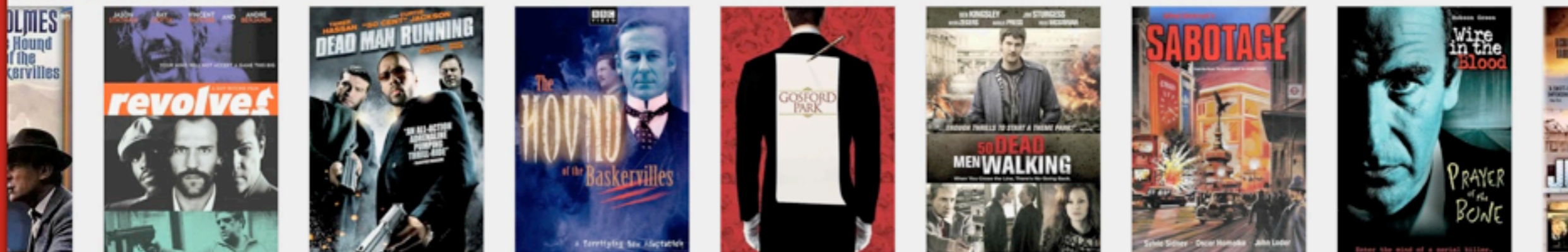
Movies, TV shows, actors, directors, genres 🔍

Genres ▾ New Arrivals Starz Play Instantly to your TV

## TOP 10 FOR BILL



## BRITISH CRIME THRILLERS



## CRITICALLY-ACCLAIMED DARK ACTION &amp; ADVENTURE



# Experience & Experimentation First

# Why start with experience?

Stay honest & pure by having experience be the driver  
(not what your boss thinks or what looks good on your resume or  
what the loudest one in the room thinks)

## Remember...

Use before you reuse

Let the experience drive the engineering

Reuse is an engineering optimization. Use is what users do.

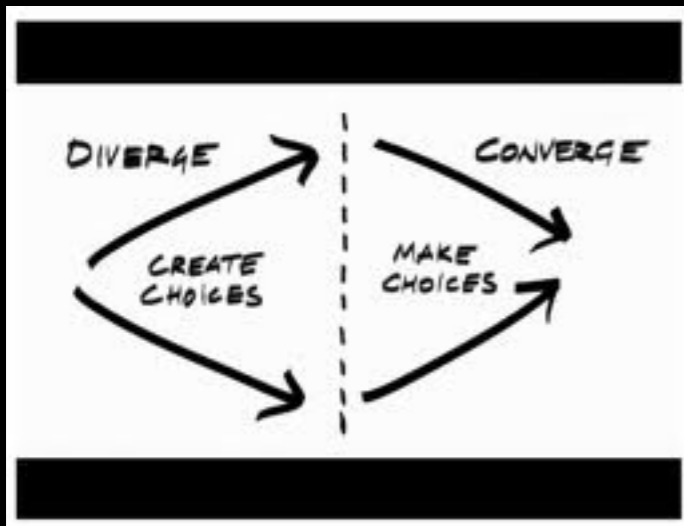
# #3 Build in Rapid Experimentation



# Build in Rapid Experimentation

Think of the UI Layer as “The Experimentation Layer”

Early rapid prototyping leads to learnings to get into the right ballpark



Follow with live A/B Testing. Lots of it.



Creates a healthy environment with constant customer feedback loops  
Contrast this with “Long Shelf Life” culture

# Lean UX: Key to Rapid Iteration



Before

Most sales of TurboTax at tax season led to conservative culture  
One major initiative a year.

Now

Test over 500 different changes in a 2 1/2 month tax season.  
Running up to seventy different tests per week.  
Make a change live on its website on Thursday, run it over the weekend, read the results on Monday, and come to conclusions starting Tuesday; then they rebuild new tests

# Lean UX: Key to Rapid Iteration

# Lean UX: Key to Rapid Iteration

Lean UX - co-located



# Lean UX: Key to Rapid Iteration

Lean UX - co-located



Product/Design team



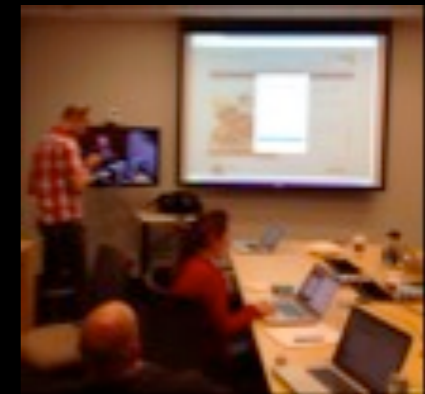
Whiteboard  
to code

UIEs



Code to  
Usability

Usability/Customers



And back  
again...

# Lean UX: Key to Rapid Iteration

Lean UX - co-located



Product/Design team



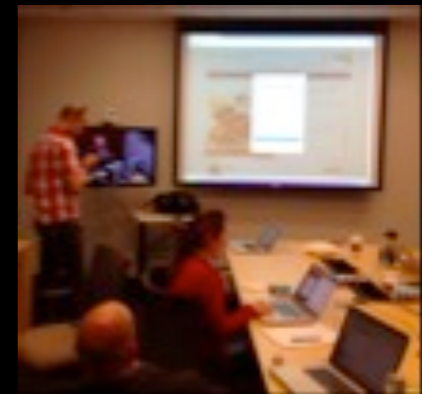
Whiteboard  
to code

UIEs



Code to  
Usability

Usability/Customers




And back  
again...

Remove the walls between teams

Use shared understanding instead of documentation

Make living code the design

Everyone has skin in the game

A large pile of unsorted, multi-colored LEGO bricks (red, blue, yellow, white, black) is shown. A black text box with a white border is overlaid on the right side of the image, containing the text "#4 Reuse Where Possible".

**#4 Reuse Where Possible**

# Requirements for Lean UI Stack

Independent of the backend language

Flexible enough to run in either the server or in the client

Equally good at building web sites as it is building web applications

Pushable outside of the application stack (publish model)

Cleanly separated from the backend/app code (ideally by JSON)

Utilize what is common to developers in the valley (what they use at night)

Quick & easy to build & tear down components

# Example of Lean Stack

## Bootstrap, from Twitter



BACKBONE.JS

UNDERSCORE.JS



{dust}



**github**  
SOCIAL CODING



# Startups move at Lean Speed

Meebo - Power of Python/Django  
Spawned Instagram, GameClosure, Clover  
Acquired by Google 2012

Rypple - Rapid prototyping/Starbucks Usability  
Acquired by Salesforce 2011

Togetherville - Many Pivots  
Acquired by Disney 2010

BagCheck - Mobile first, get it live early, build on open source stack  
Acquired by Twitter 2011

# Engineering Lean

It's not all reuse. It is the experience.

Stop re-inventing what has already been invented

Github is your best friend. Make reuse your bootstrap.

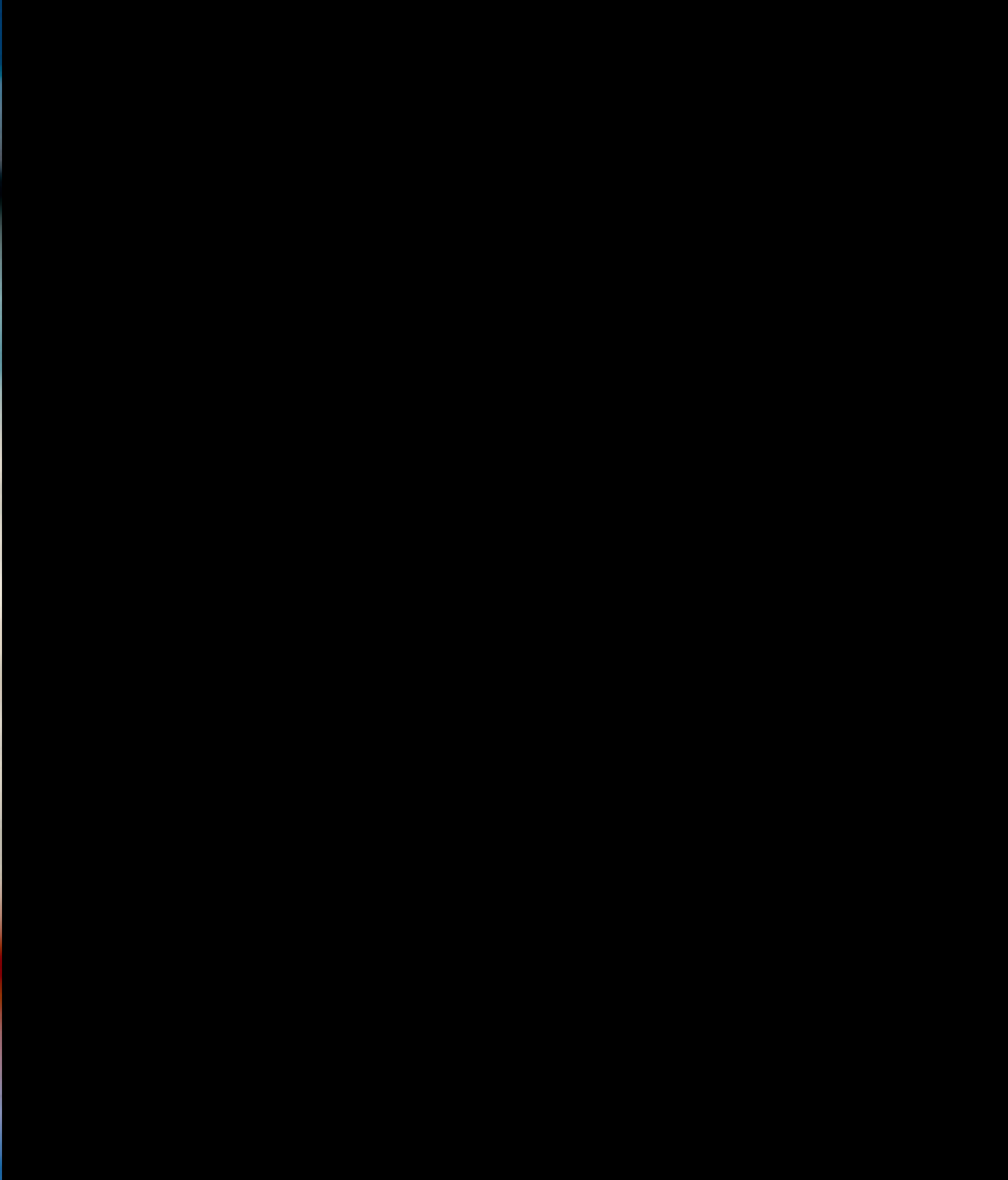
Remember Zuck's Law

"If I had to build this in a garage, how would I build it?"

The bottom line: get the user feedback loop flowing and keep it flowing.

# **The Mission**

Bringing Design to Life  
Quickly



## Presentation

[billwscott.com/share/presentations/2012/ow/](http://billwscott.com/share/presentations/2012/ow/)

## Blogs

<http://looksgoodworkswell.com>

<http://designingwebinterfaces.com>

## Follow me on twitter!

@billwscott



## Book

<http://designingwebinterfaces.com>

## Picture Credits (Creative Commons)

<http://www.flickr.com/photos/neilsingapore/4047105116/sizes/l/>

[http://www.flickr.com/photos/smb\\_flickr/439040132/](http://www.flickr.com/photos/smb_flickr/439040132/)

<http://www.flickr.com/photos/therevsteve/3104267109/sizes/o/>

<http://www.flickr.com/photos/st3f4n/4193370268/sizes/l/>

<http://www.flickr.com/photos/eole/380316678/sizes/z/>

<http://www.flickr.com/photos/cobalt/3035453914/sizes/z/>

<http://www.flickr.com/photos/mbiskoping/6075387388/>

<http://www.flickr.com/photos/fragglawker/2370316759/sizes/z/>

<http://www.flickr.com/photos/soldiersmediacenter/4685688778/sizes/z/>