Building Clouds

Pivotal™

Who am I?

What we do

How we do it

Who am I?

What we do

How we do it

Who am I?

What we do

How we do it

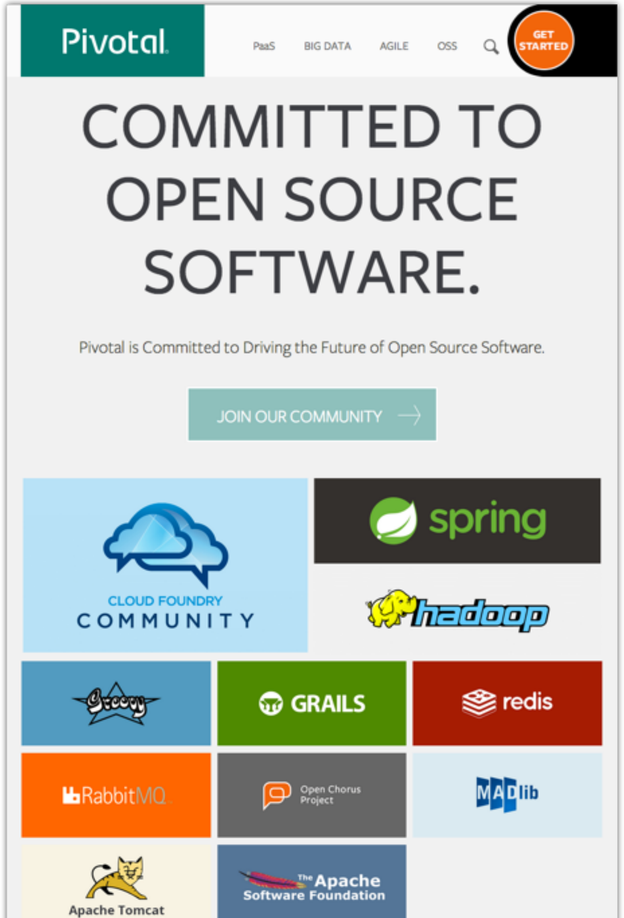
Massive Organization

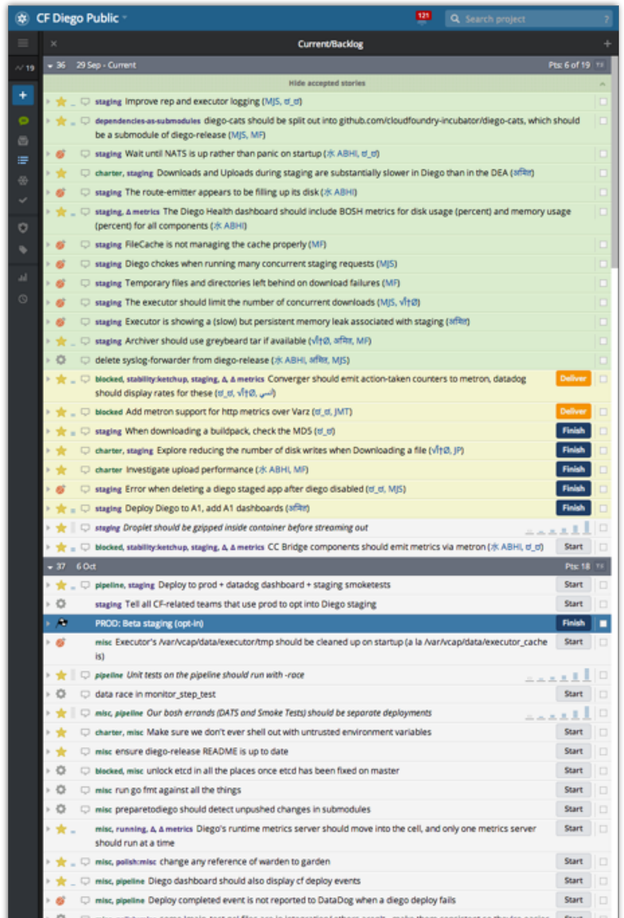
Hundreds of web applications

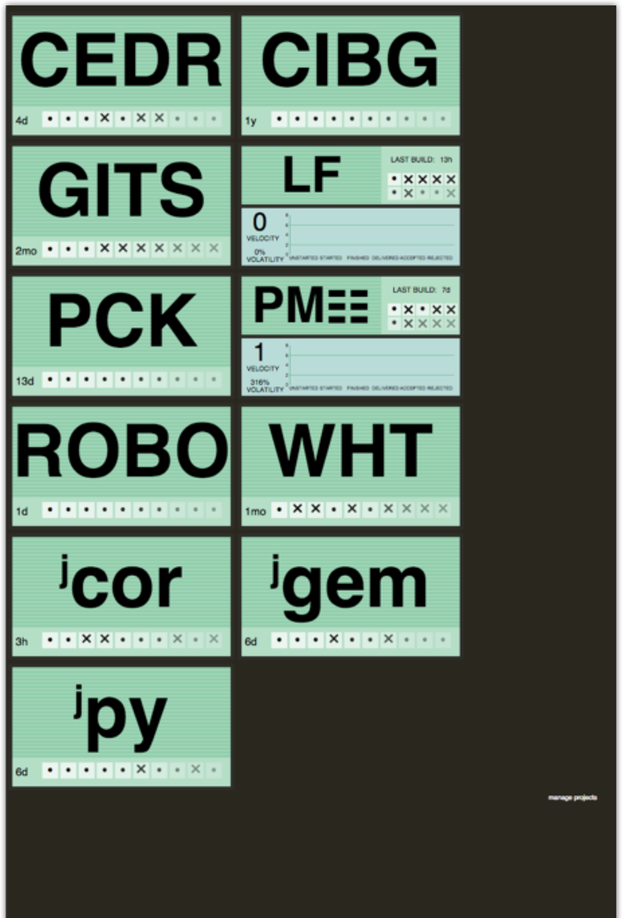
Pivotal™

Massive Organization

Hundreds of web applications



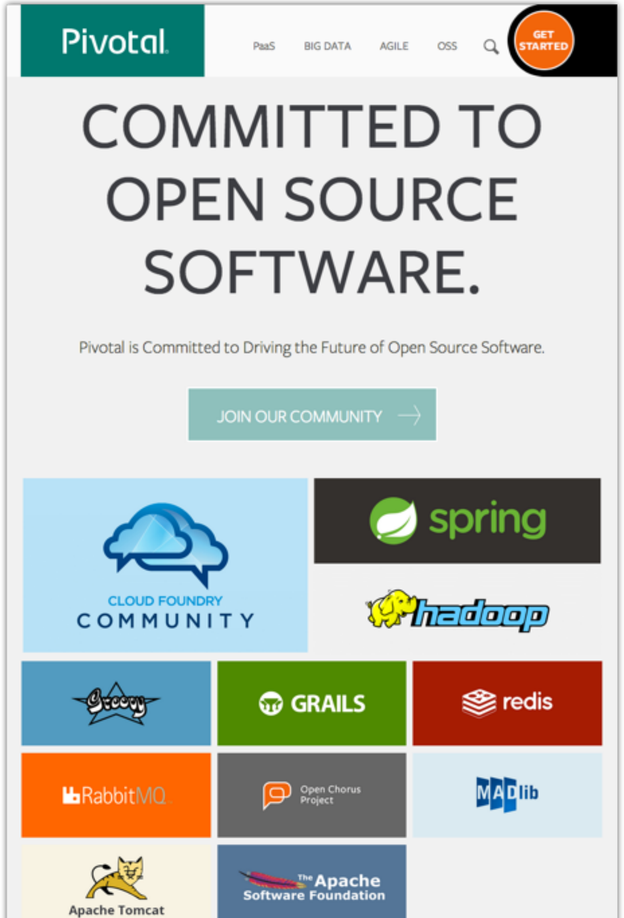


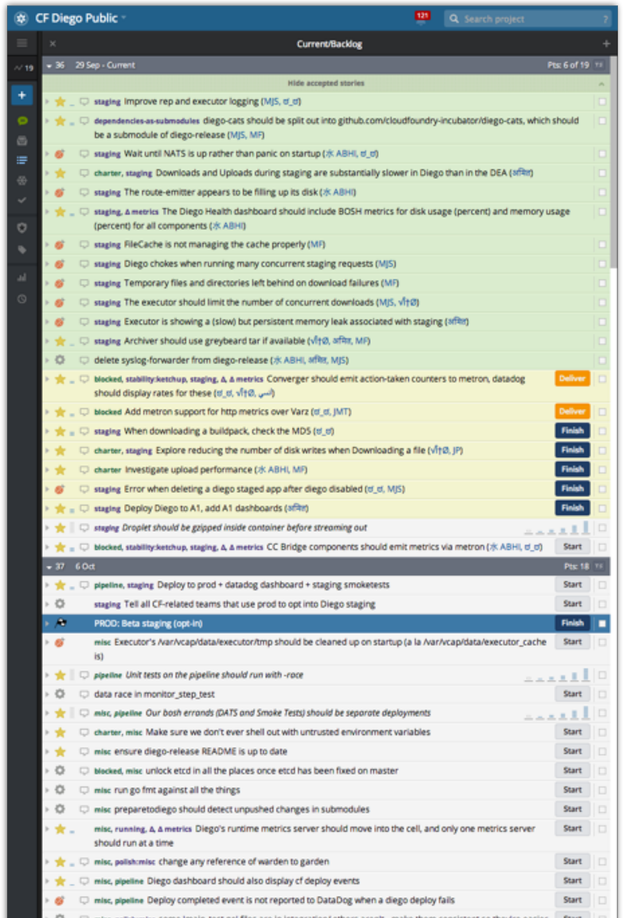


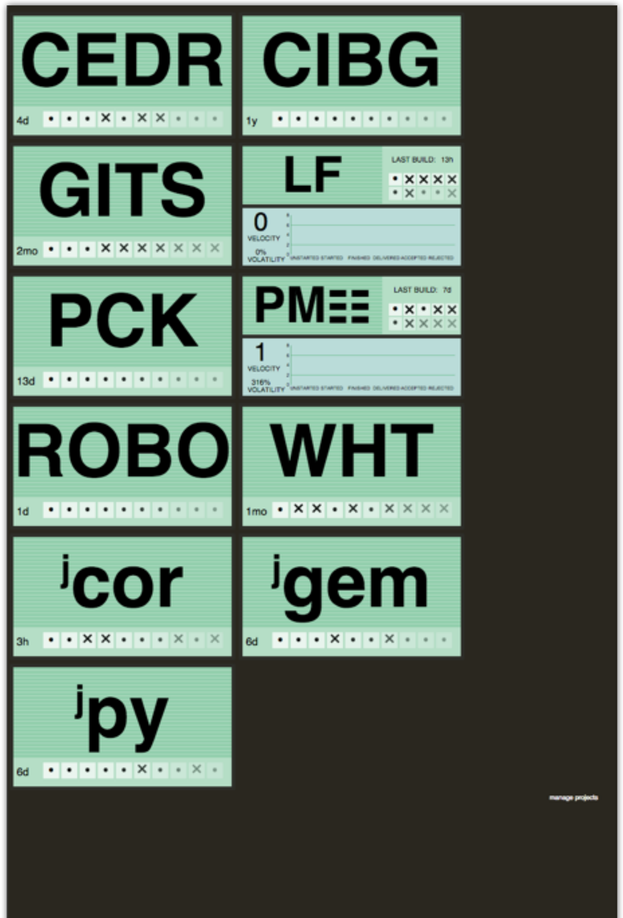
Pivotal™

Massive Organization

Hundreds of **complex** web applications





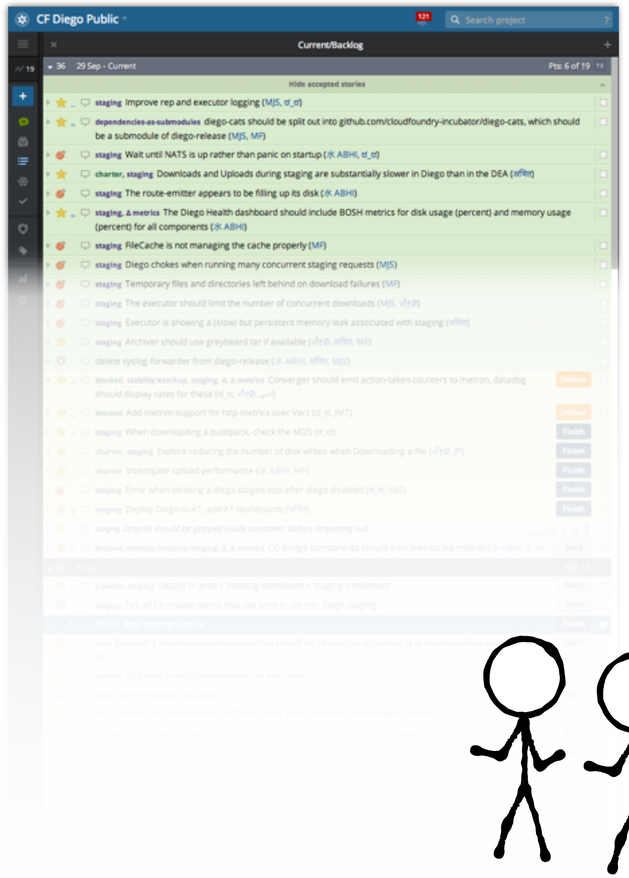


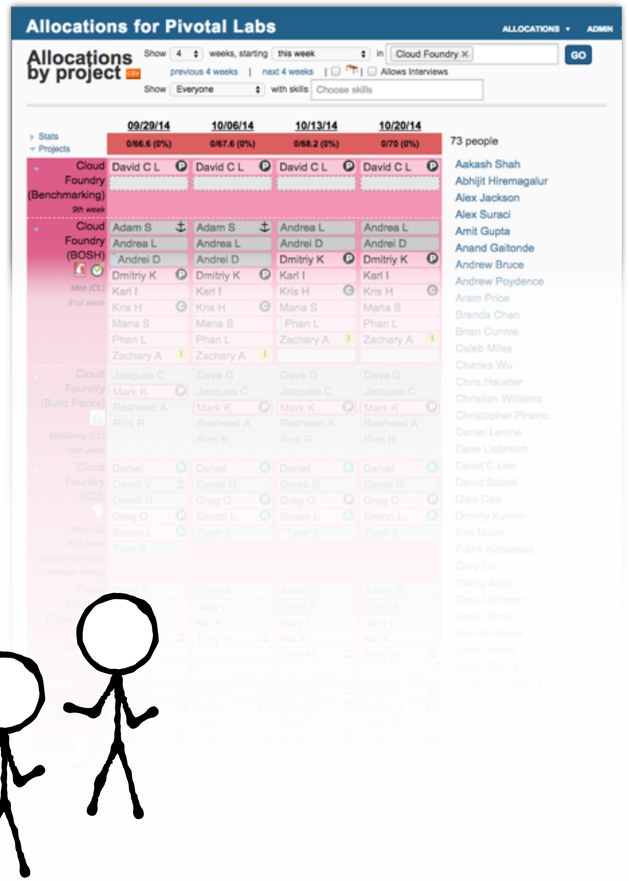
Pivotal™

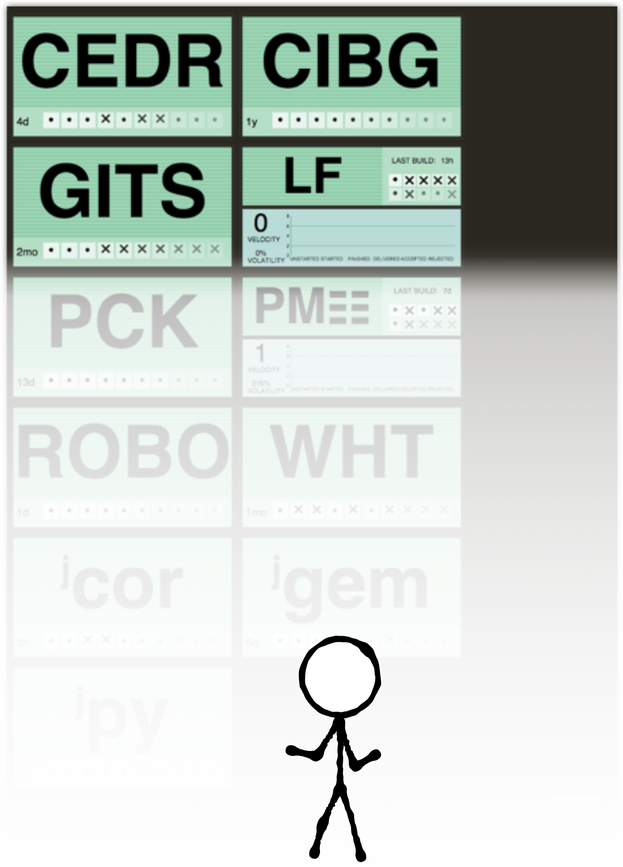
Built by many teams of developers

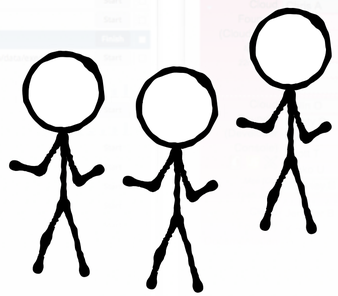
Massive Organization

Hundreds of web applications









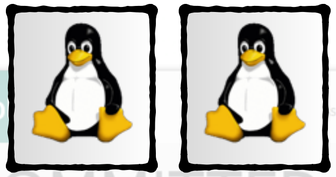
Pivotal™

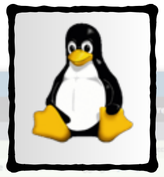
Massive Organization

Hundreds of web applications

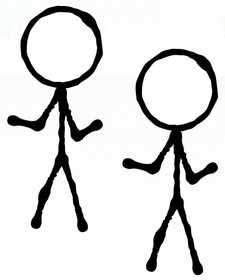
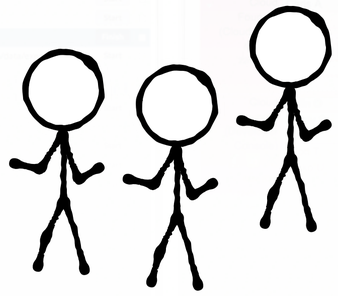
Built by many teams of developers

Running on many different servers








Pivotal™

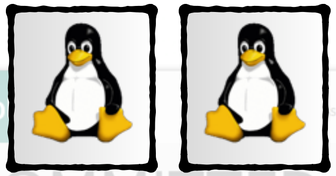
Massive Organization

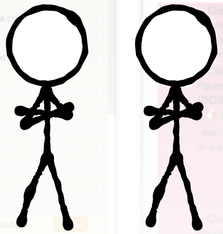
Hundreds of web applications

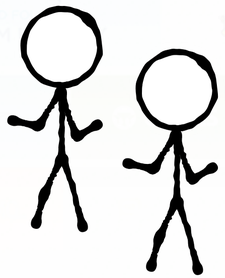
Built by many teams of developers

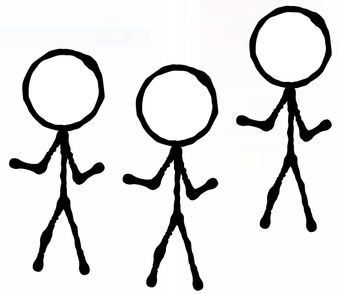
Running on many different servers

Managed by many different sysadmins









Pivotal™

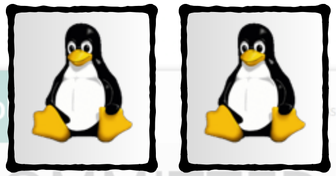
Massive Organization

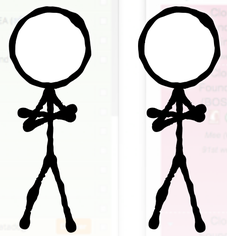
Hundreds of web applications

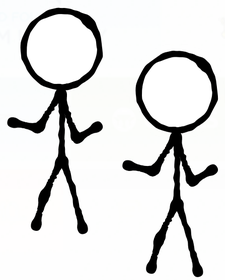
Built by many teams of developers

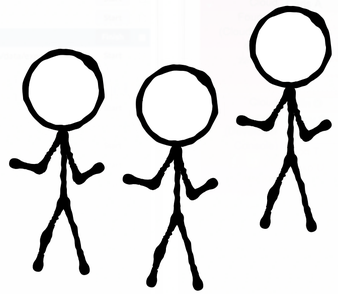
Running on many different servers

Managed by many different sysadmins

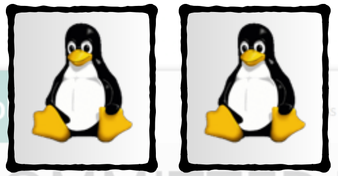


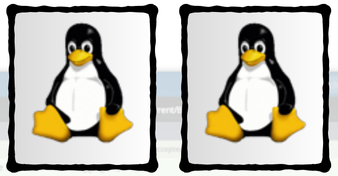


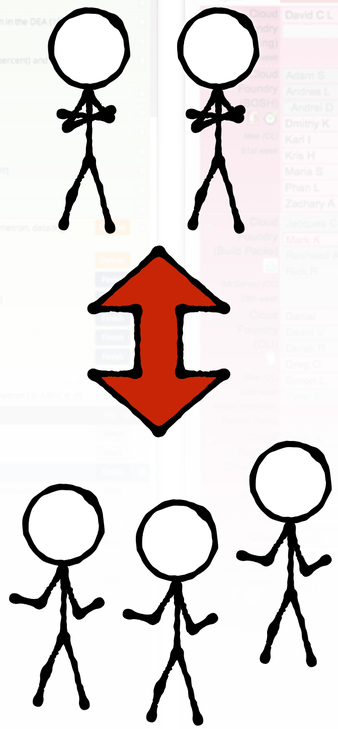




Source of conflict and miscommunication

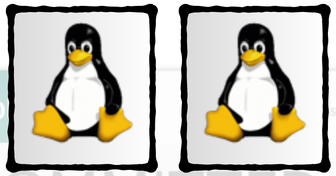


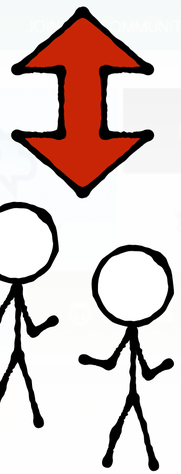


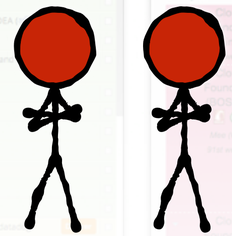


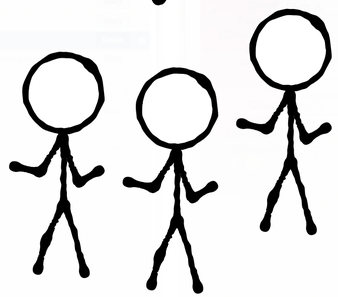
Source of conflict and miscommunication

Overworked sysadmins



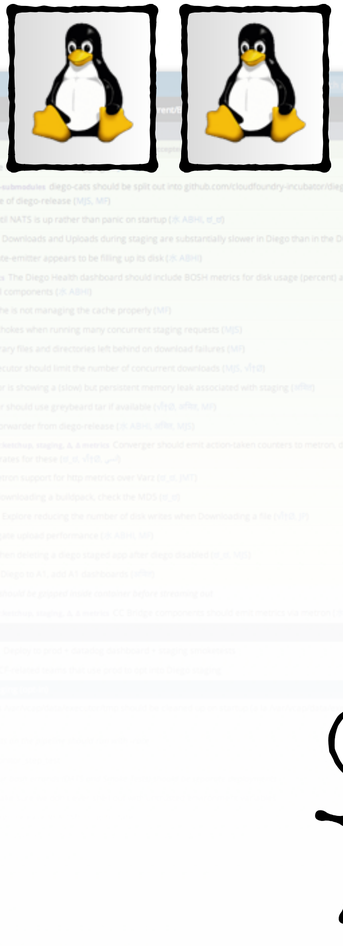


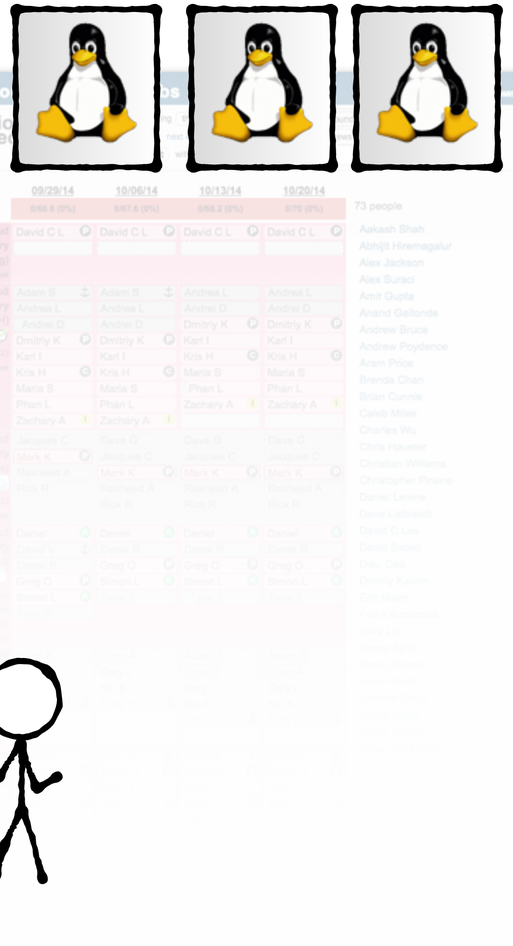




Source of conflict and miscommunication

Overworked sysadmins

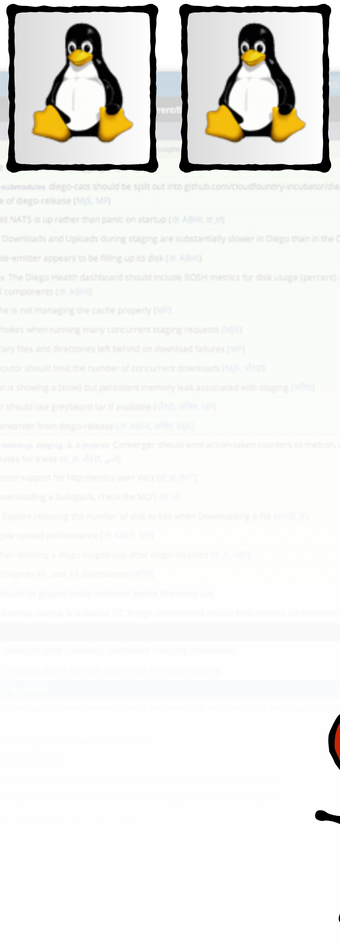


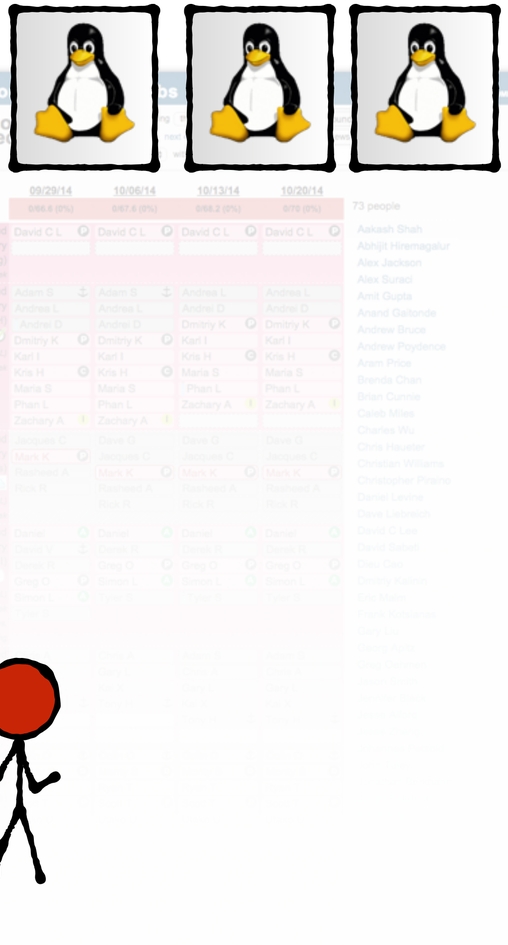


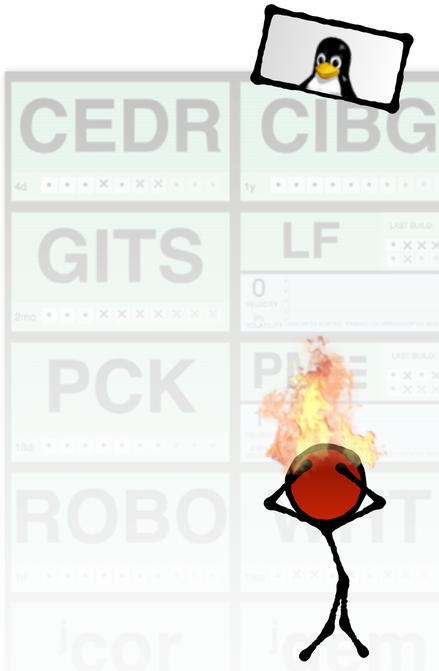
Source of conflict and miscommunication

Overworked sysadmins

Inefficient, blocked, developers trying to update their apps





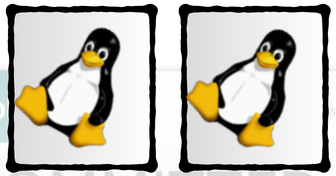


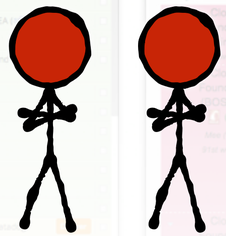
Source of conflict and miscommunication

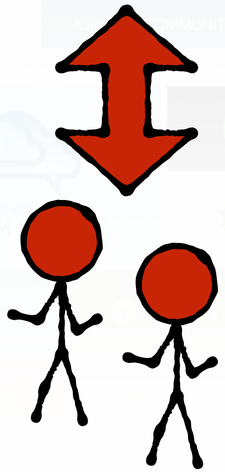
Overworked sysadmins

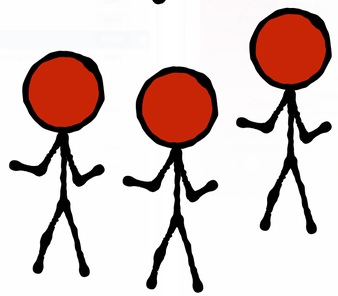
Inefficient, blocked, developers trying to update their apps

Inconsistent configurations









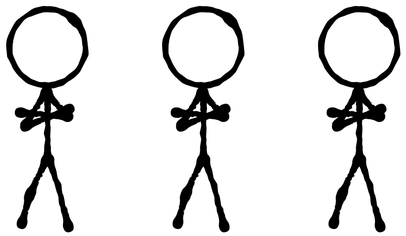
CLOUD FOUNDRY™

Platform as a Service

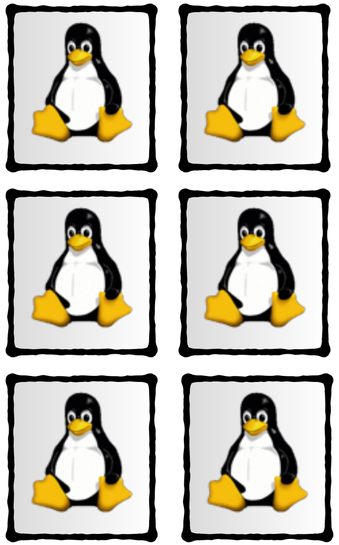
CLOUD
FOUNDRY™

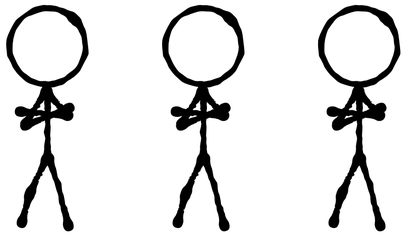
CLOUD

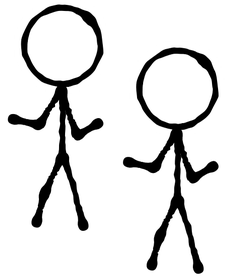
FOUNDRY™

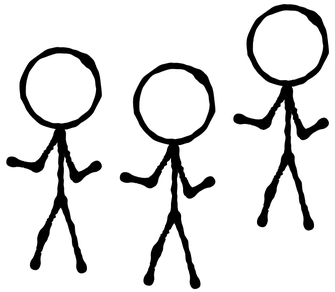


CLOUD FOUNDRY™

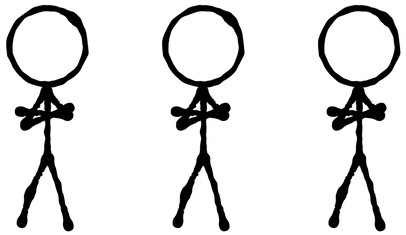


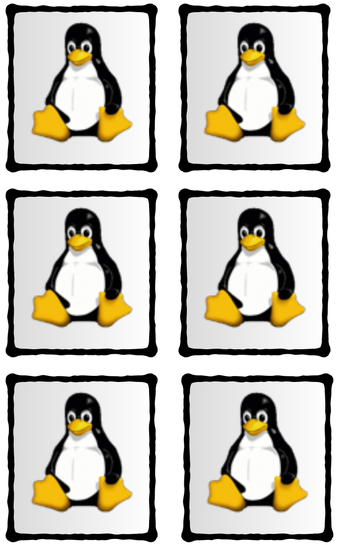


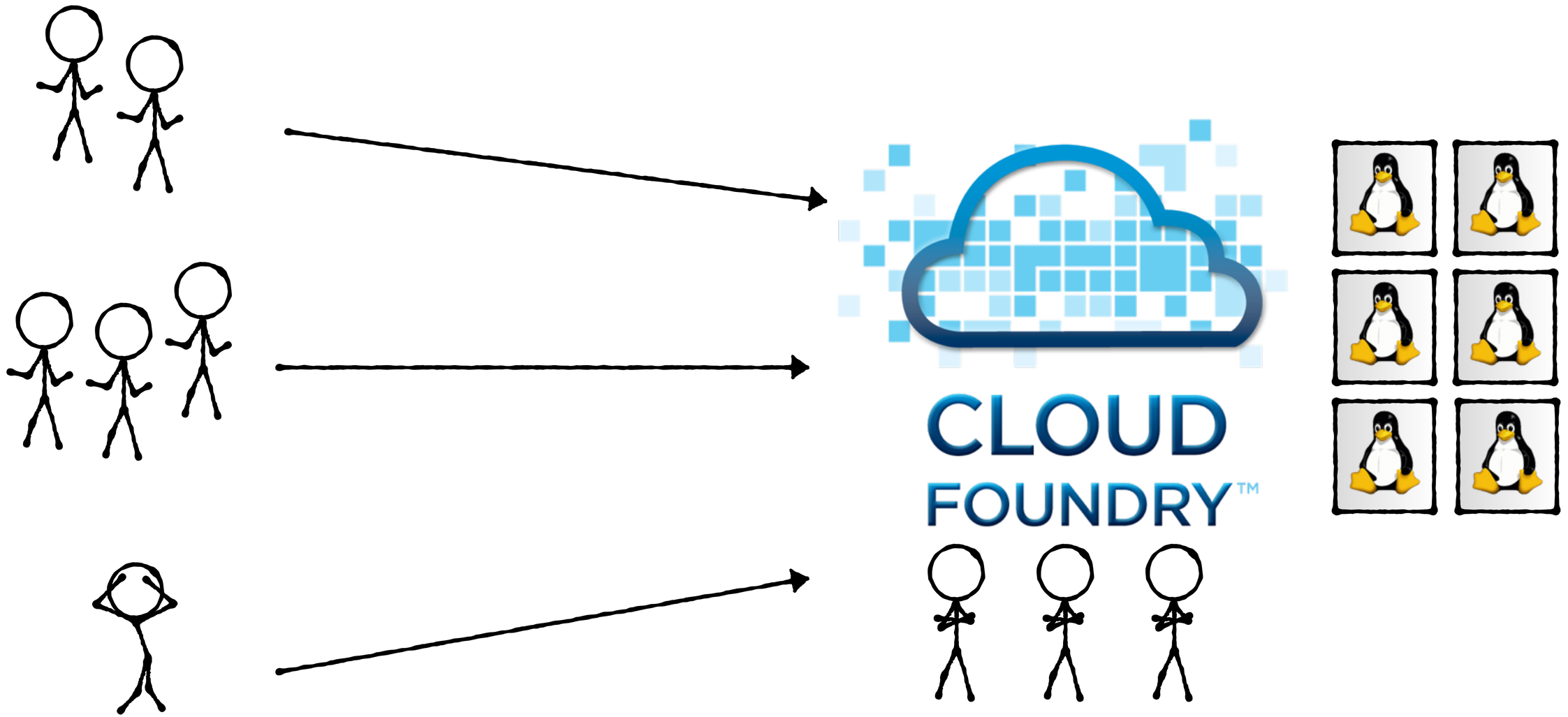




CLOUD
FOUNDRY™

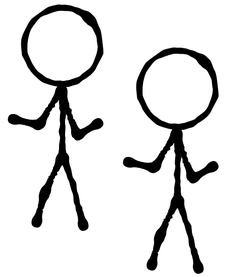


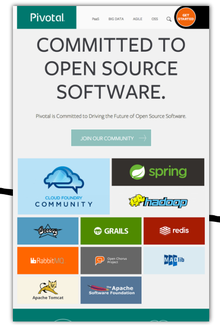


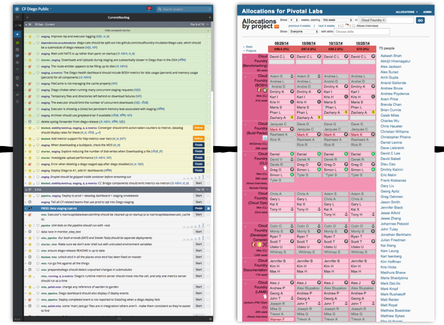


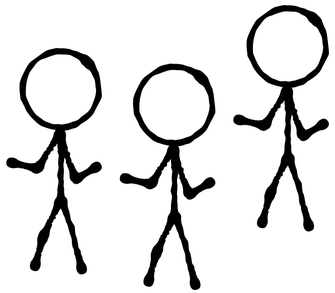

```
> cf push
```

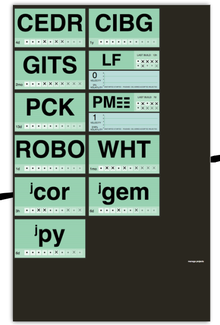
Code → Running Applications

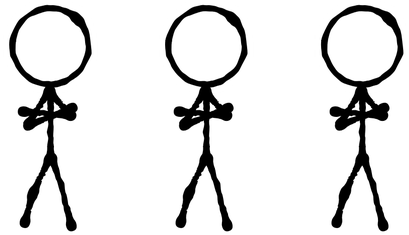


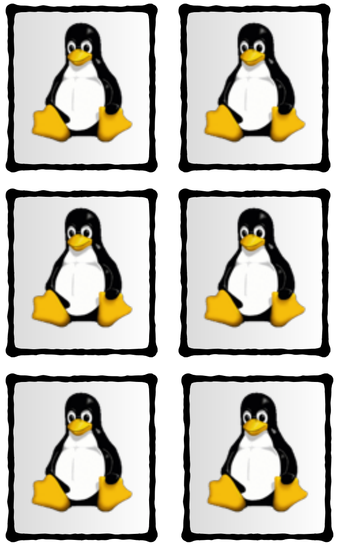






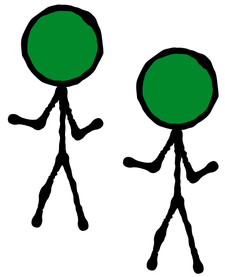


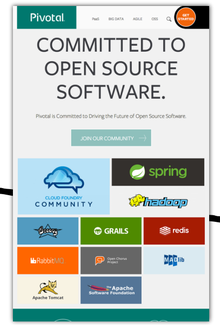


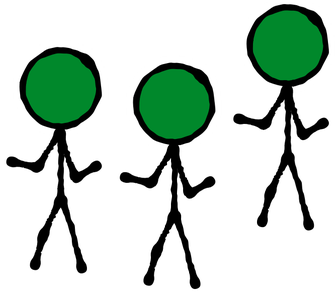


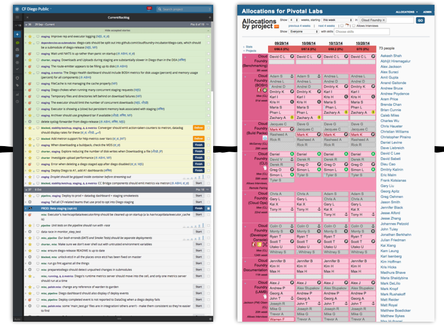
```
> cf push
```

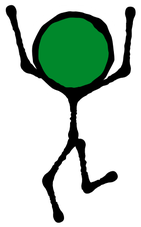
Code → Running Applications

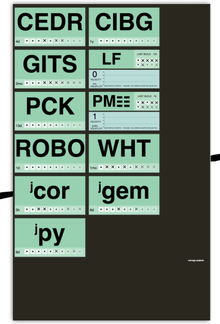




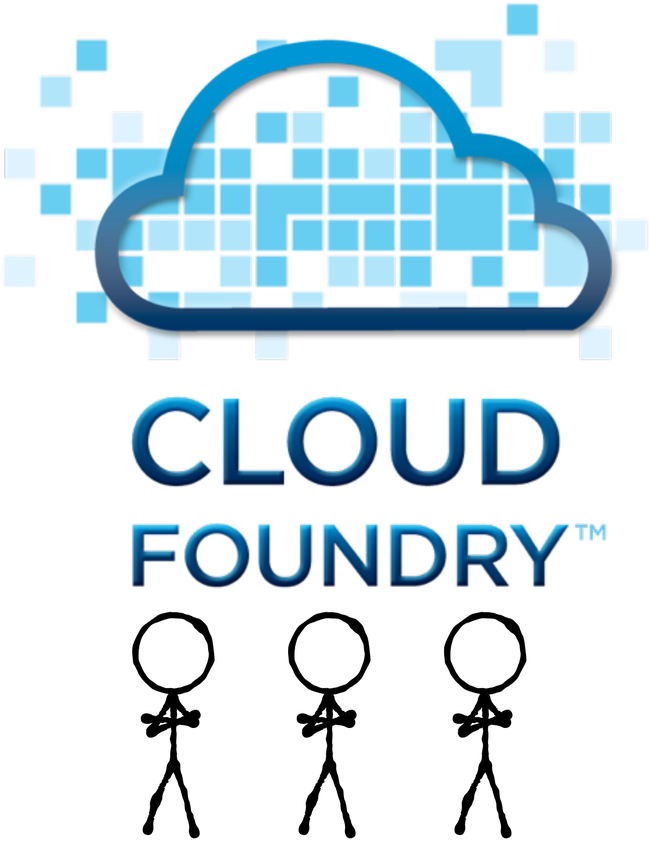


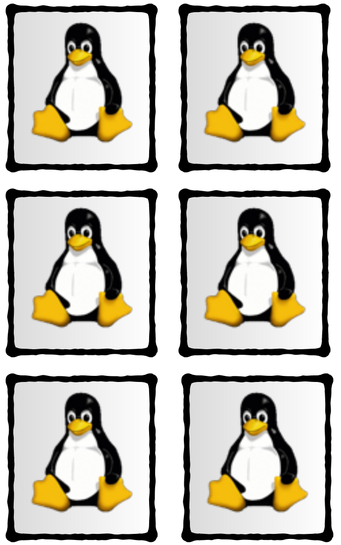




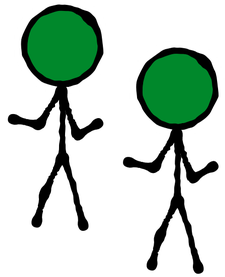


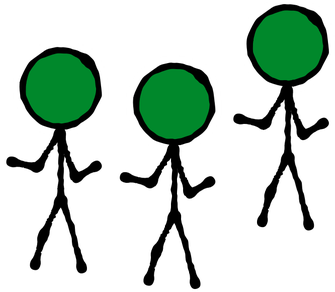
> cf push

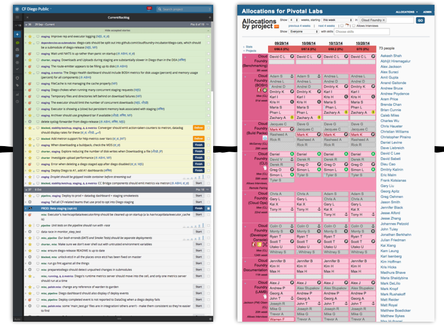


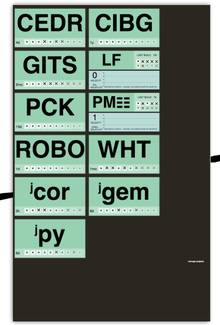


Code → Running Applications

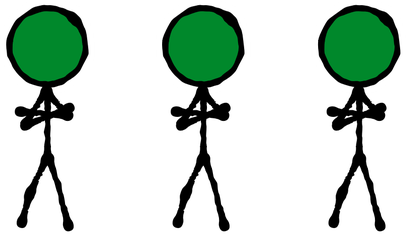


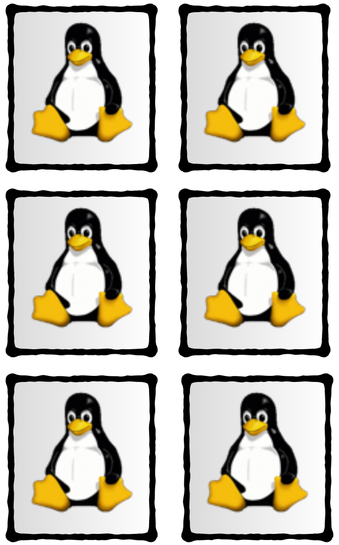






> cf push





CLOUD FOUNDRY™

Platform as a Service

CLOUD FOUNDRY™

Open source **Platform** as a Service

CLOUD FOUNDRY™

Open source **Platform** as a Service

that runs on many clouds

Who am I?

What we do

How we do it

Pivotal™

culture

(why you should work here!)

Pairing

Testing

Agile Planning

Pivotal™

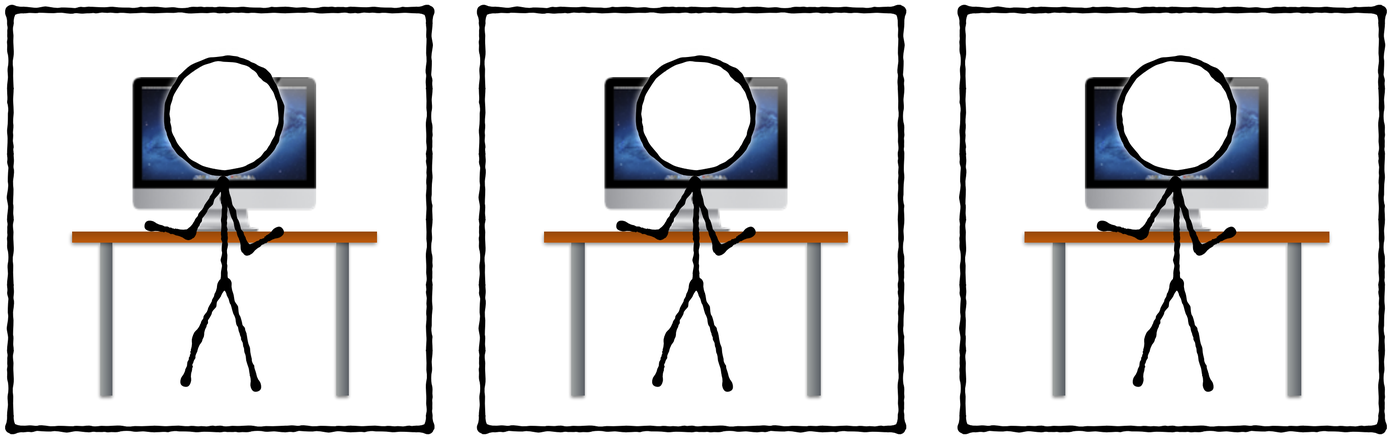
culture

(why you should work here!)

Pairing

Testing

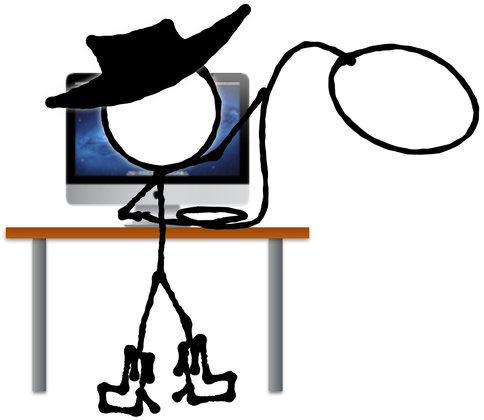
Agile Planning



Distraction

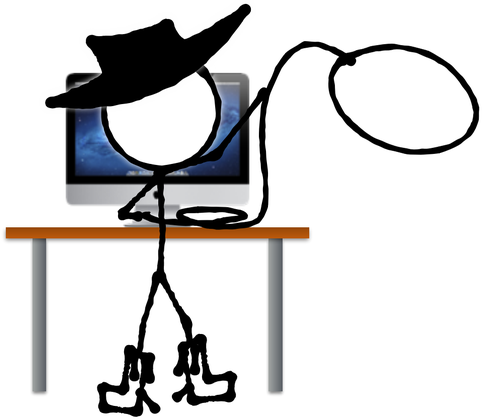
Distraction

Cowboyism



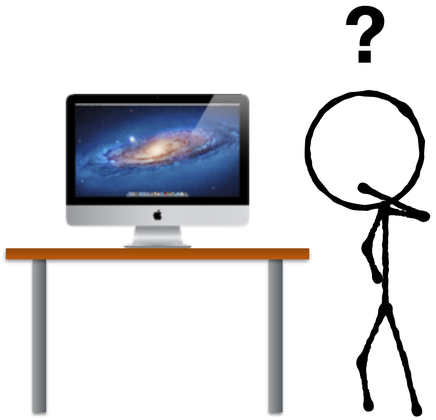
Distraction

Cowboyism



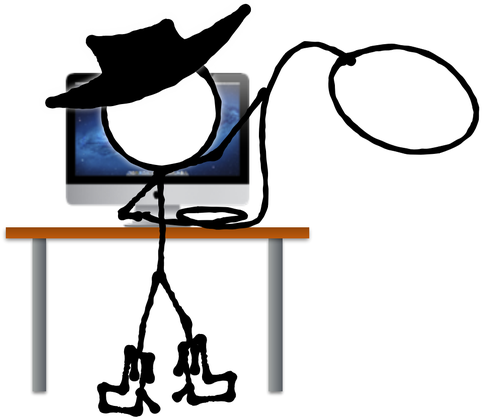
Distraction

Cowboyism



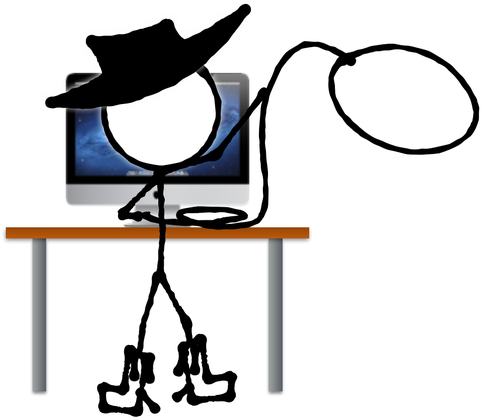
Distraction

Cowboyism



Distraction

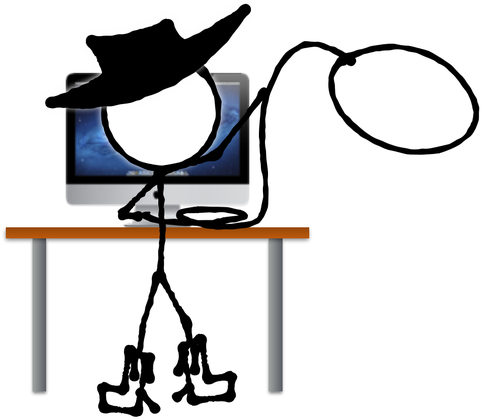
Cowboyism



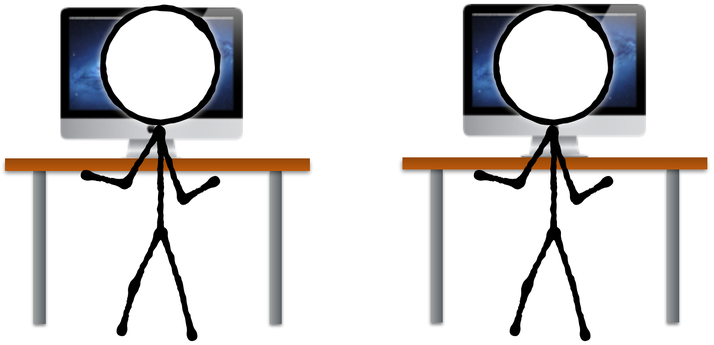
Unsutainable

Distraction

Cowboyism



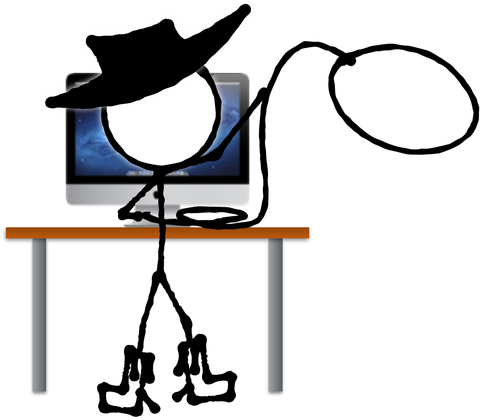
Unsutainable



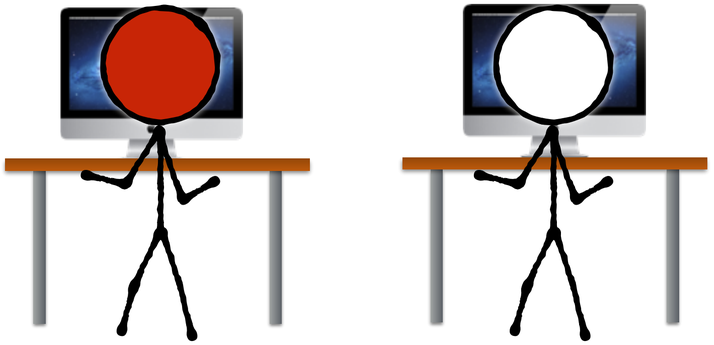
Miscommunication

Distraction

Cowboyism



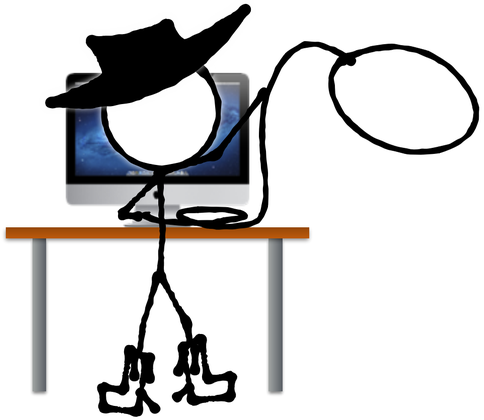
Unsutainable



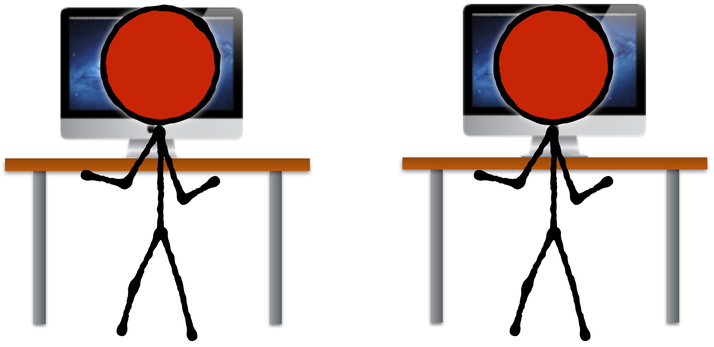
Miscommunication

Distraction

Cowboyism



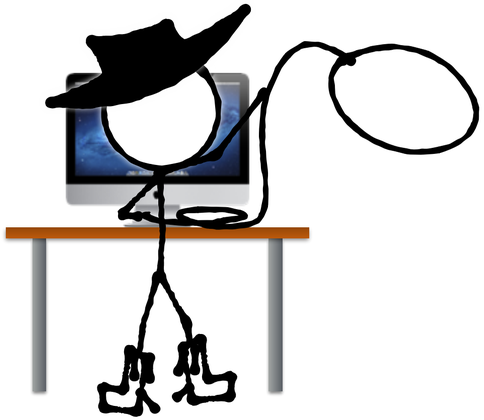
Unsutainable



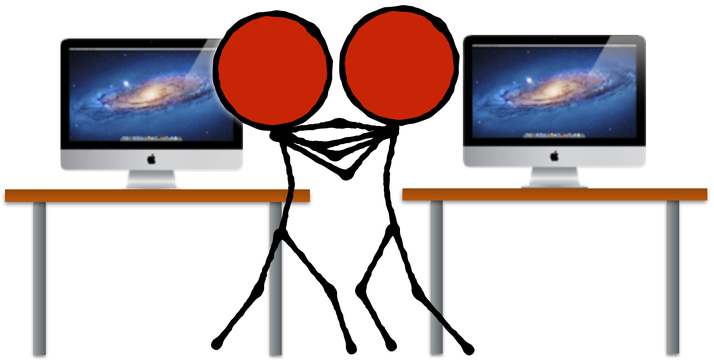
Miscommunication

Distraction

Cowboyism



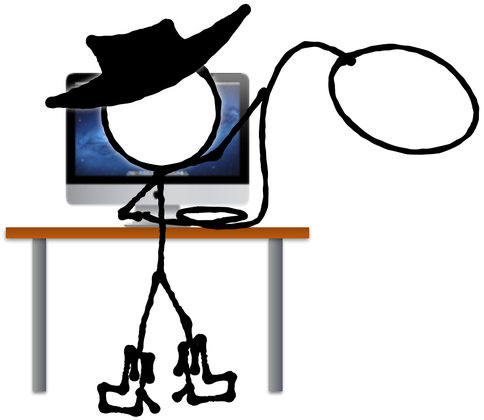
Unsutainable



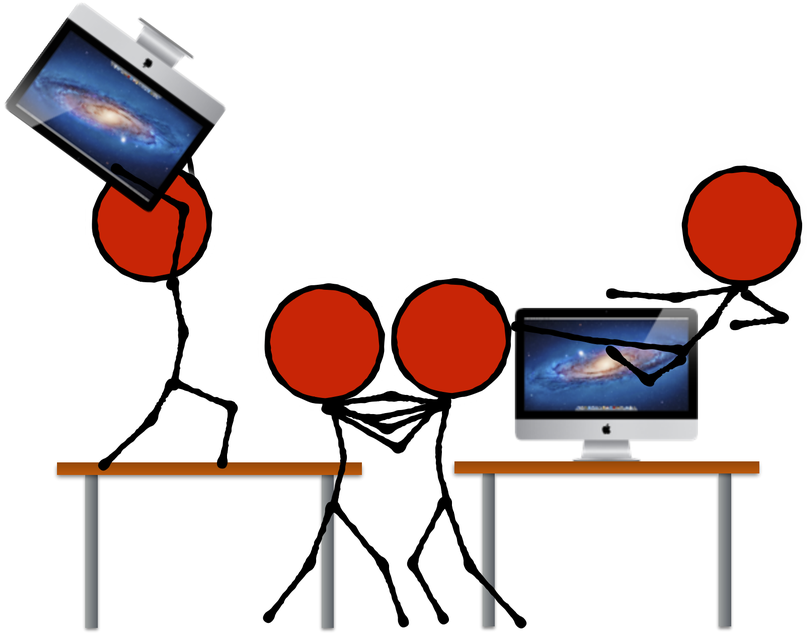
Miscommunication

Distraction

Cowboyism



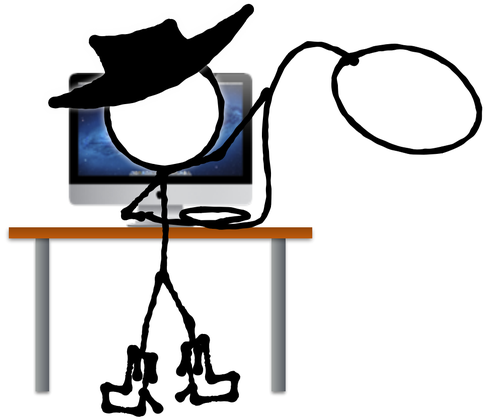
Unsutainable



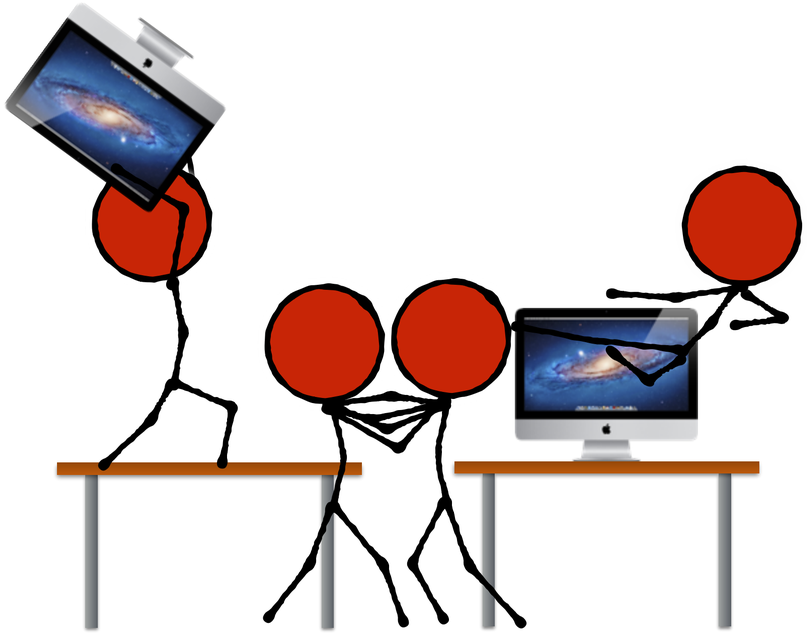
Miscommunication

Distraction

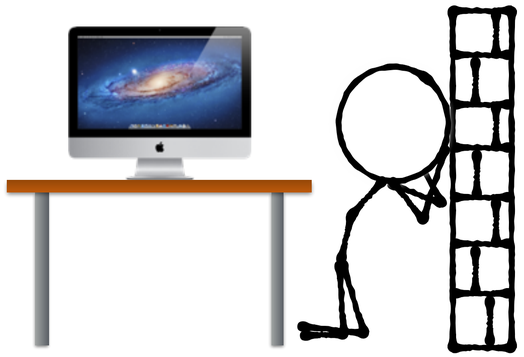
Cowboyism



Unsutainable



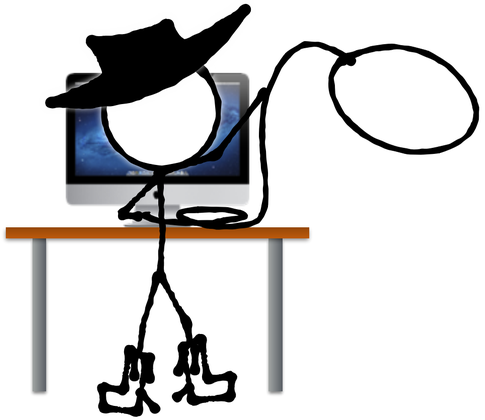
Miscommunication



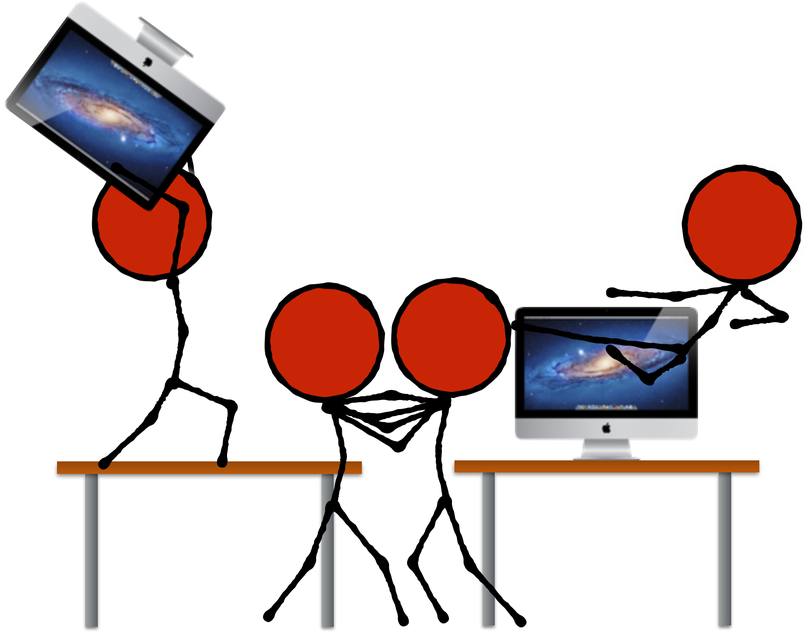
Stuck

Distraction

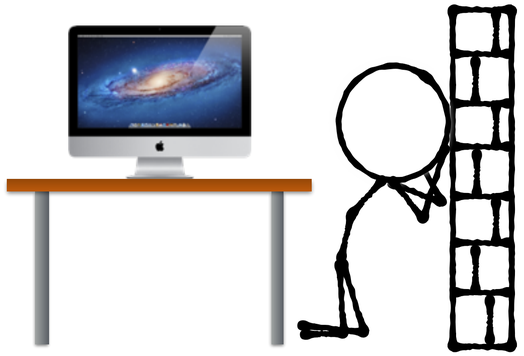
Cowboyism



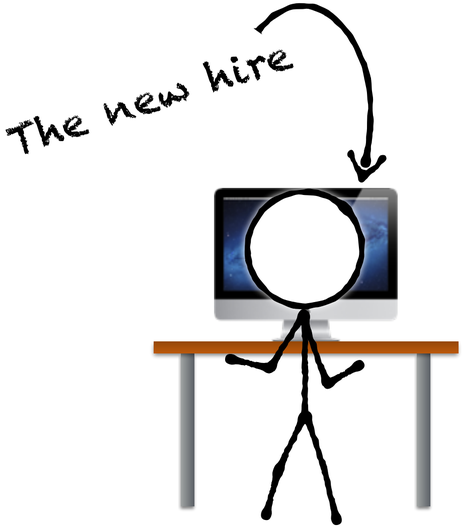
Unsutainable



Miscommunication



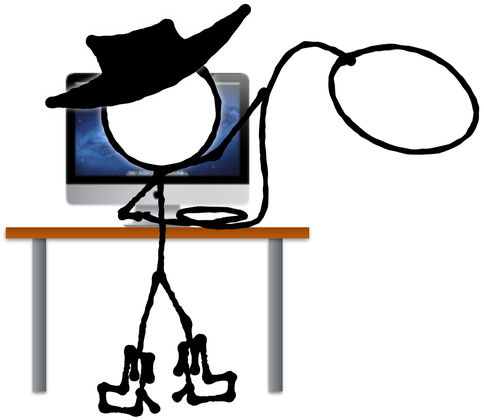
Stuck



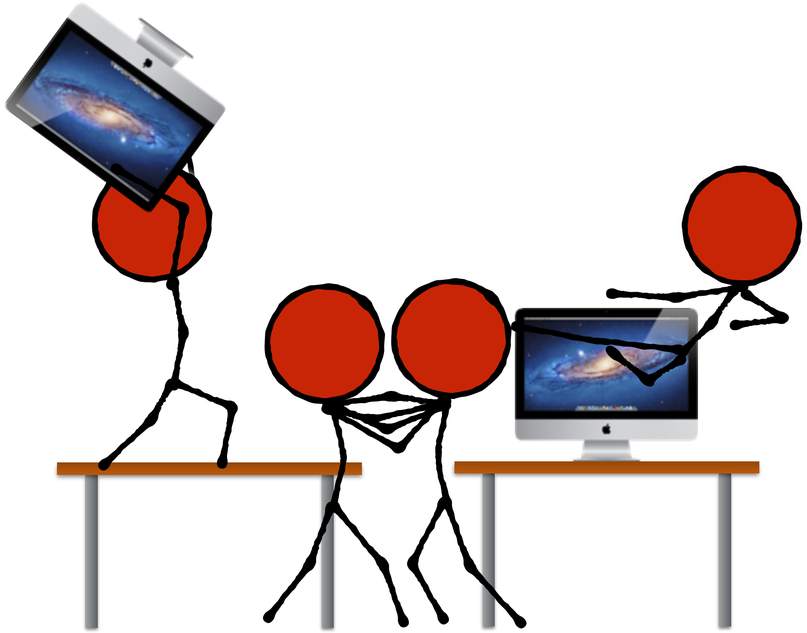
Noobs

Distraction

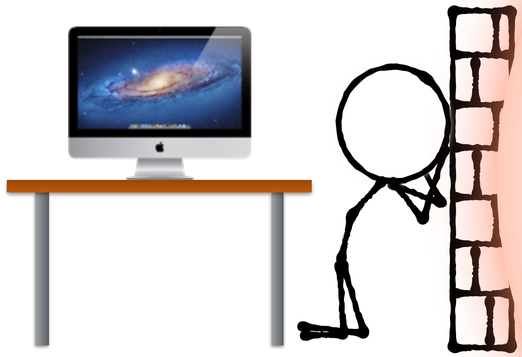
Cowboyism



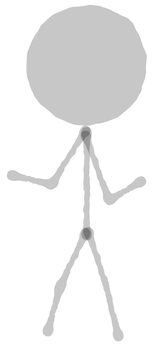
Unsutainable

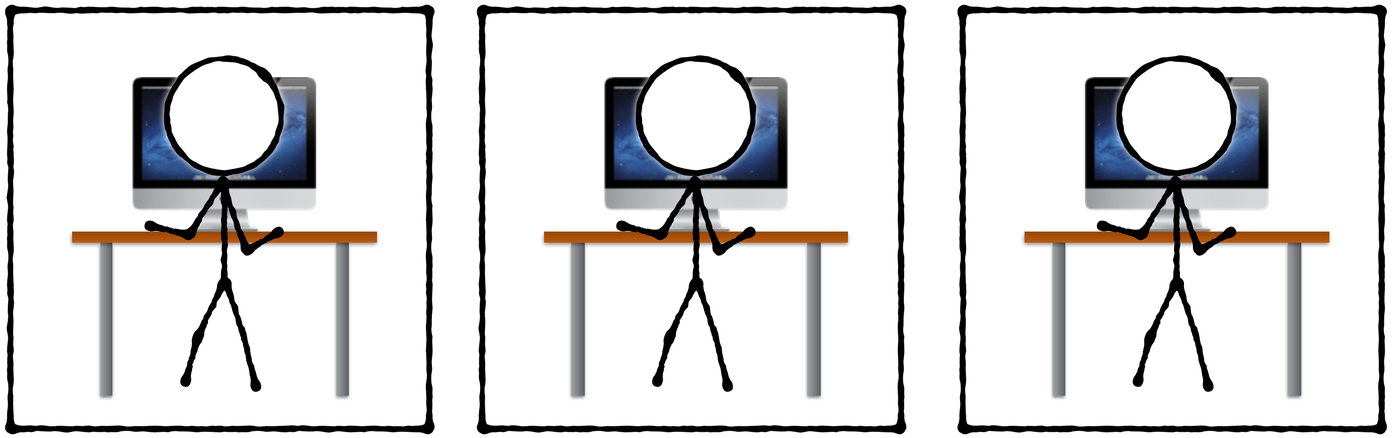


Miscommunication

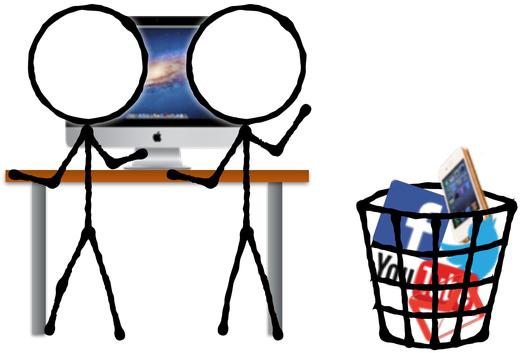


Stuck

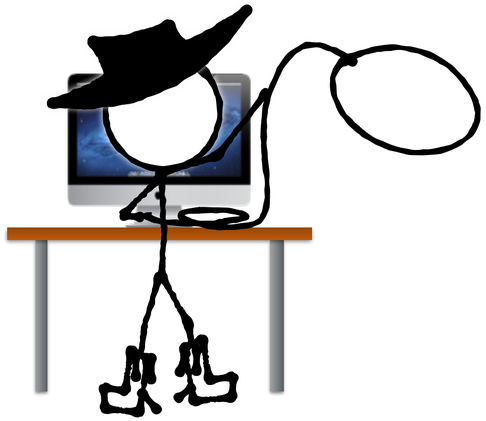




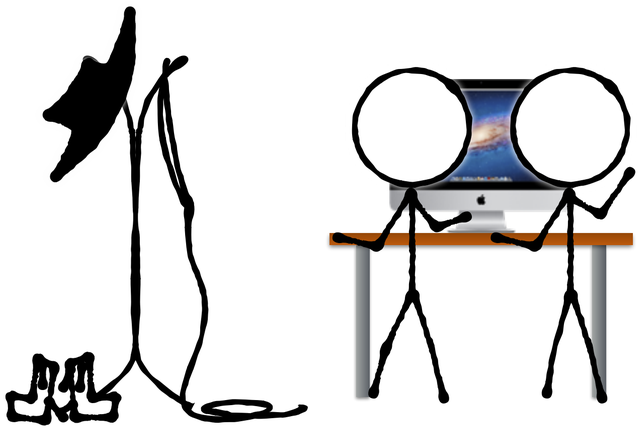
Distraction



Distraction-free

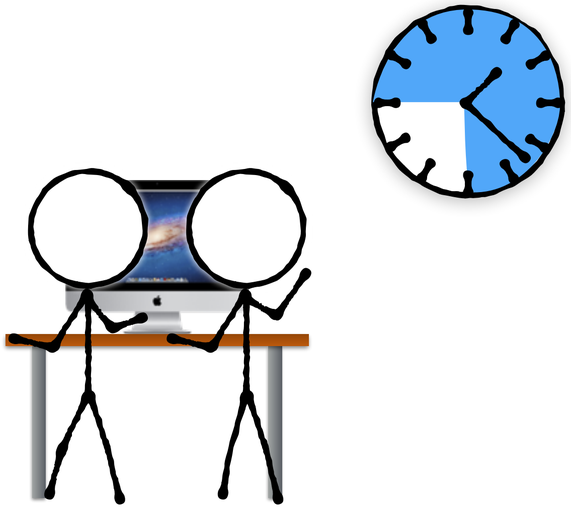


Cowboyism

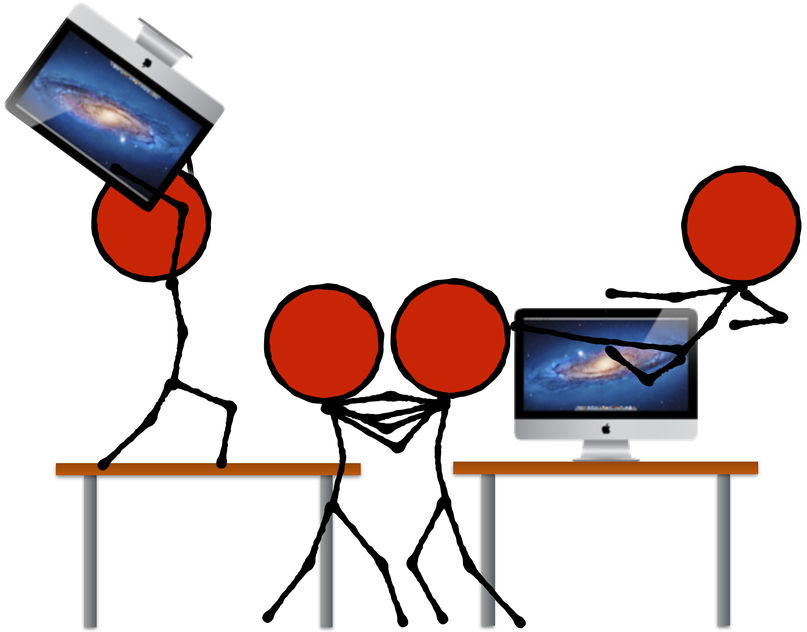


Collaboration

Unsutainable

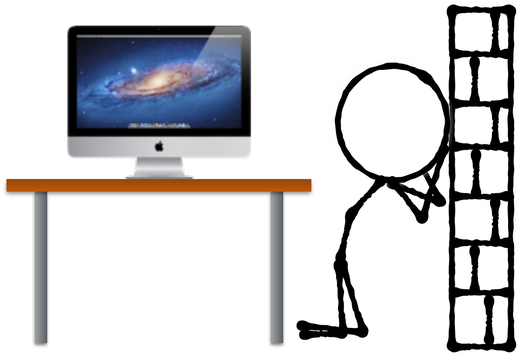


Sustainable

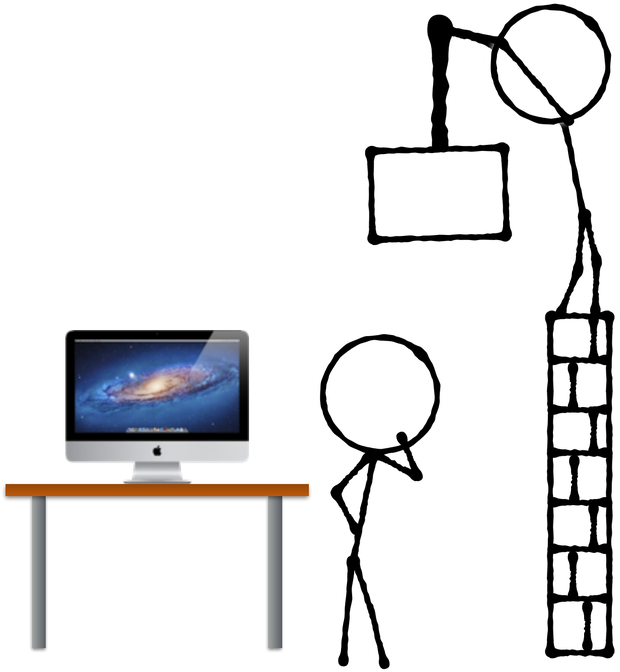


Miscommunication

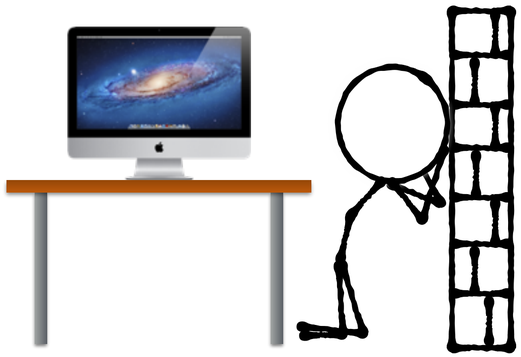
Constant Communication



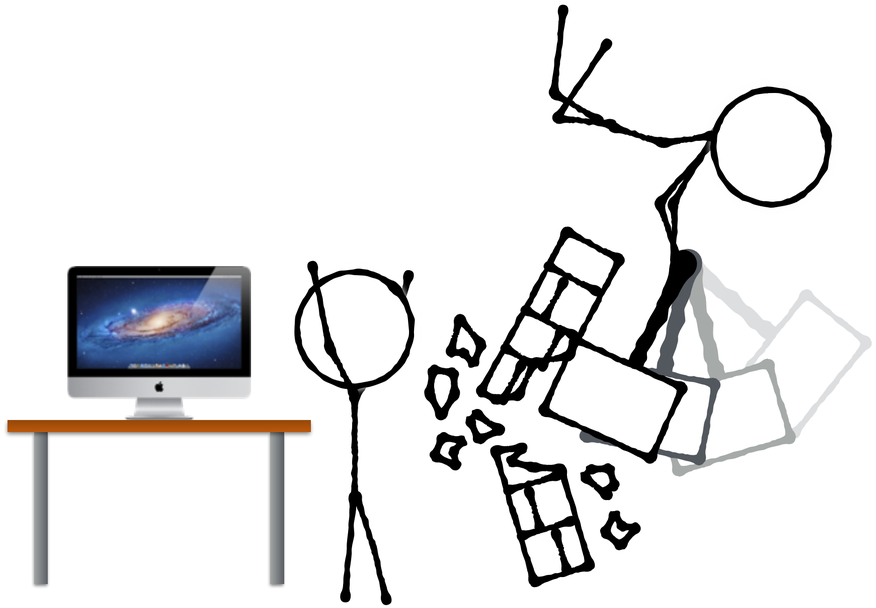
Stuck



Unstuck

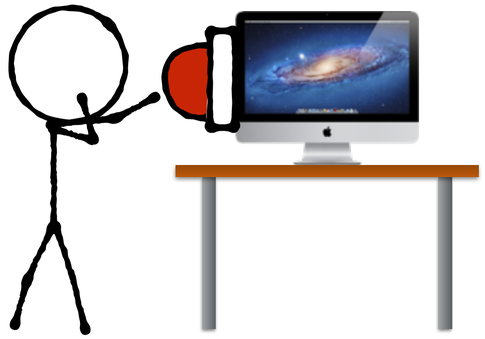


Stuck

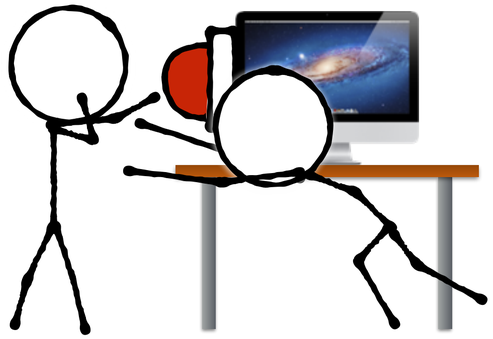


Unstuck

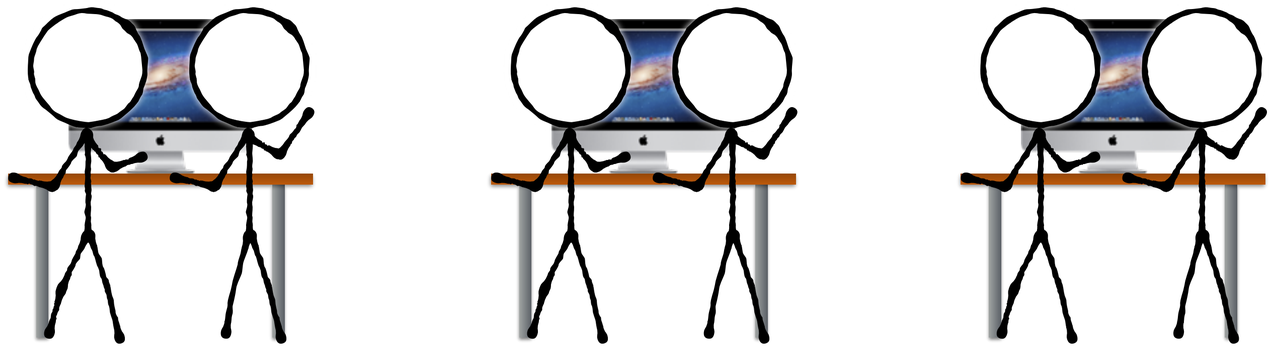
Noobs



Noobs



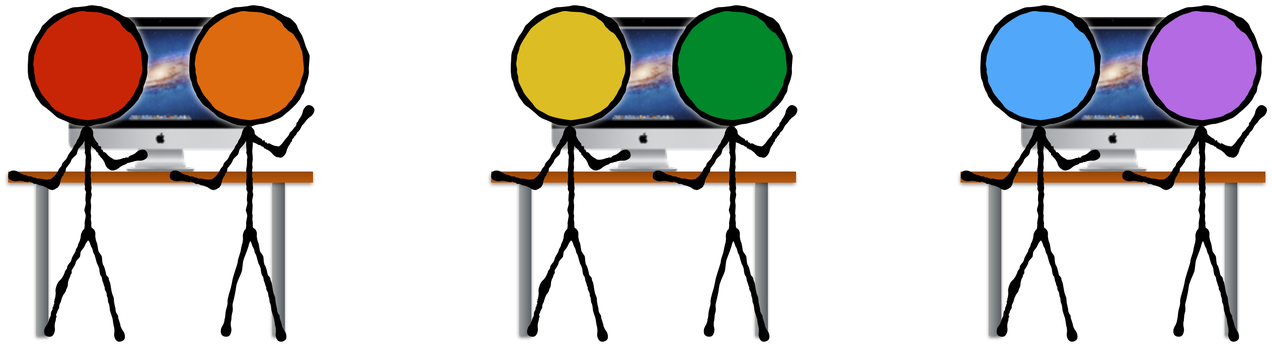
Teachers



We rotate pairs

Every day

More pairs means more possible pairings

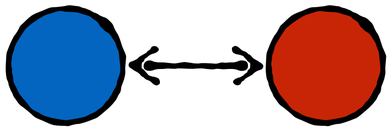


We rotate pairs

Every day

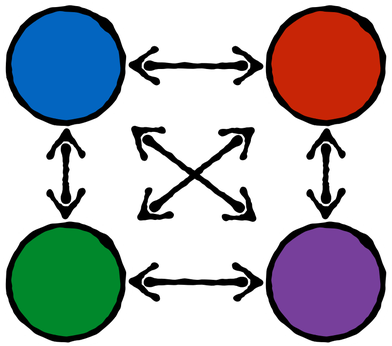
More pairs means more possible pairings

1 Pair



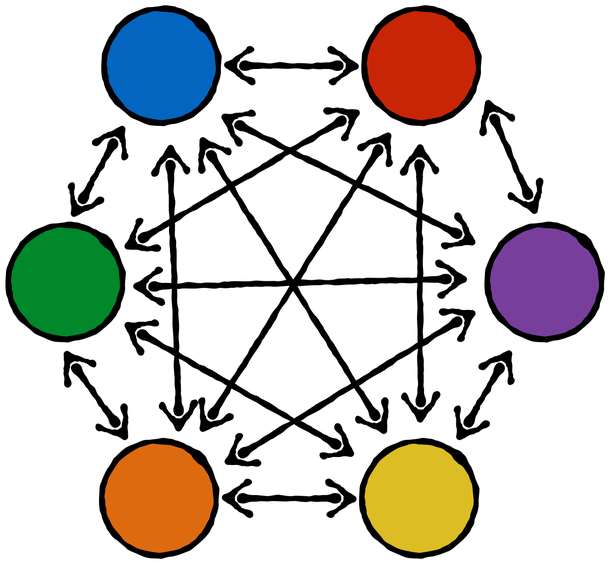
1 Pairing

2 Pairs



3 Pairings

3 Pairs



15 Pairings


KILLERSPIN
REVOLUTION

Pivotal™

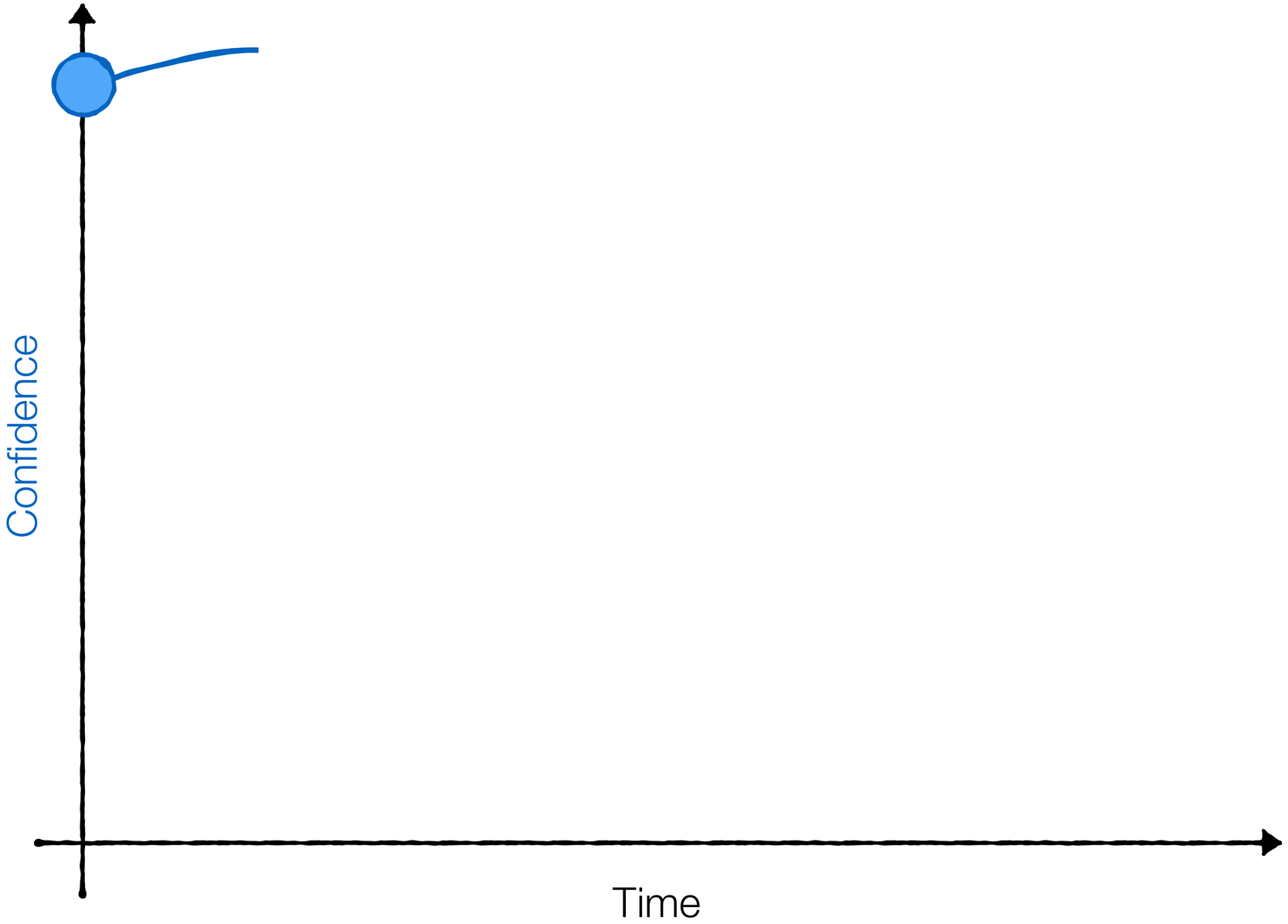
culture

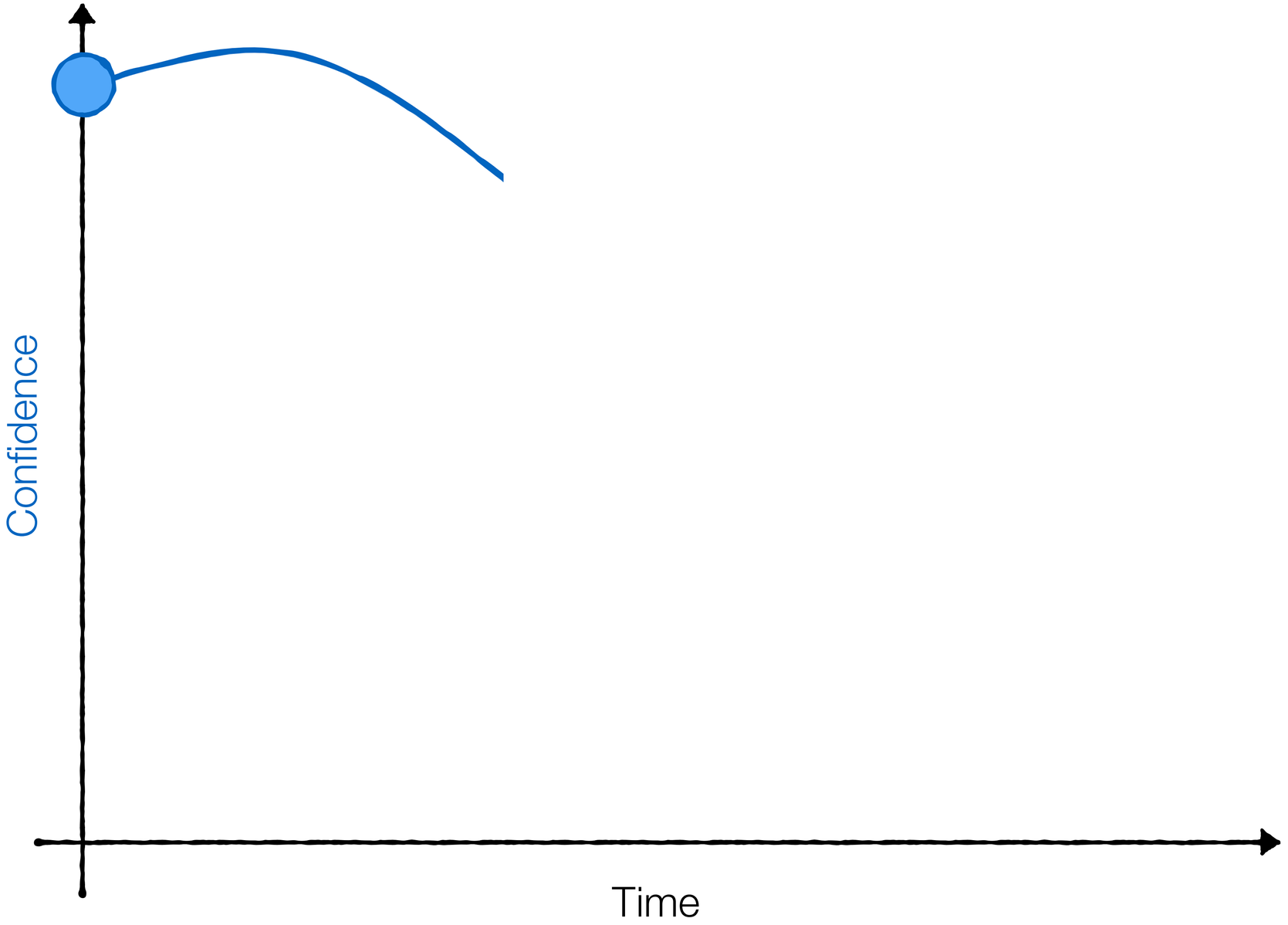
Pairing

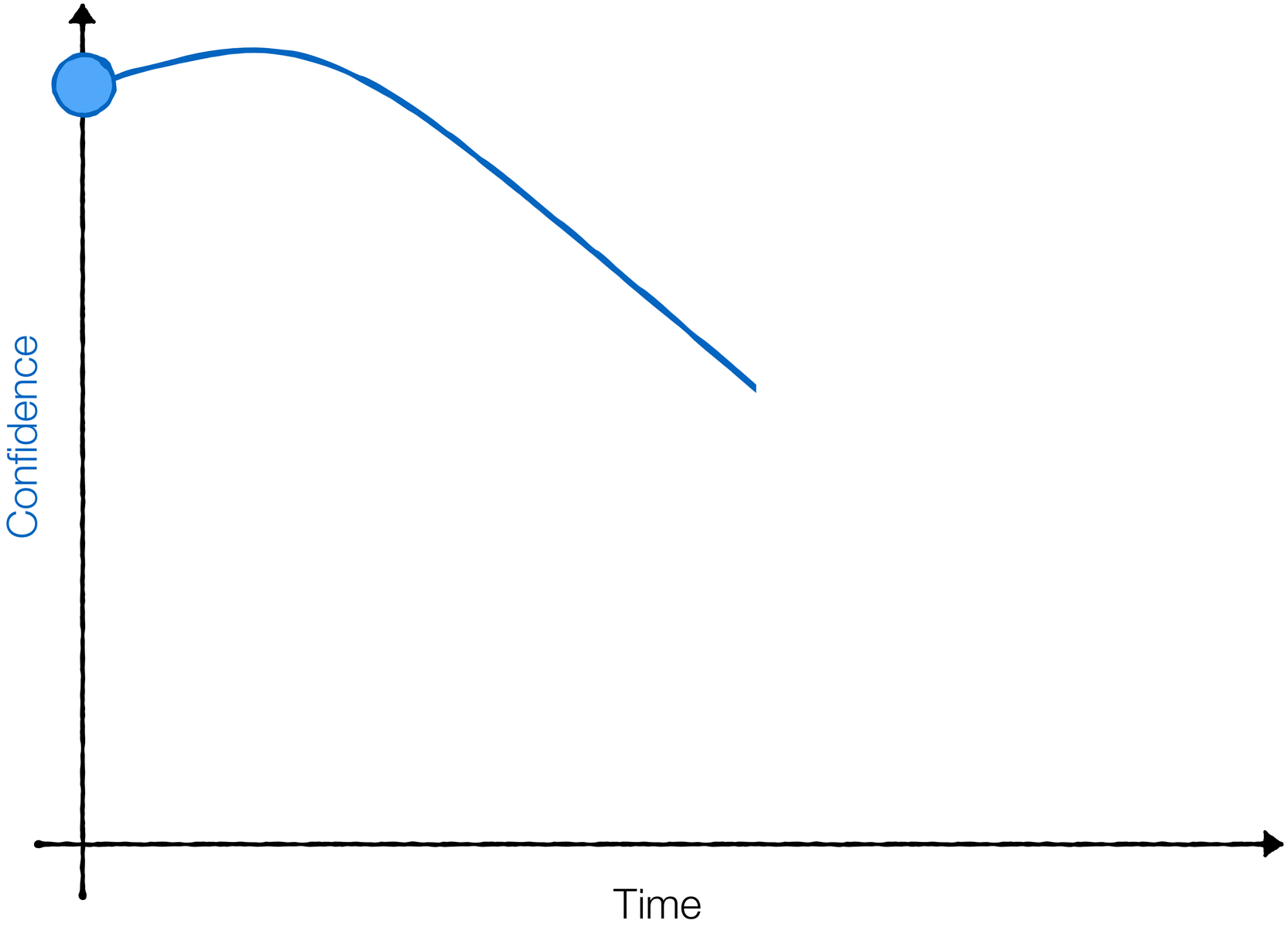
Testing

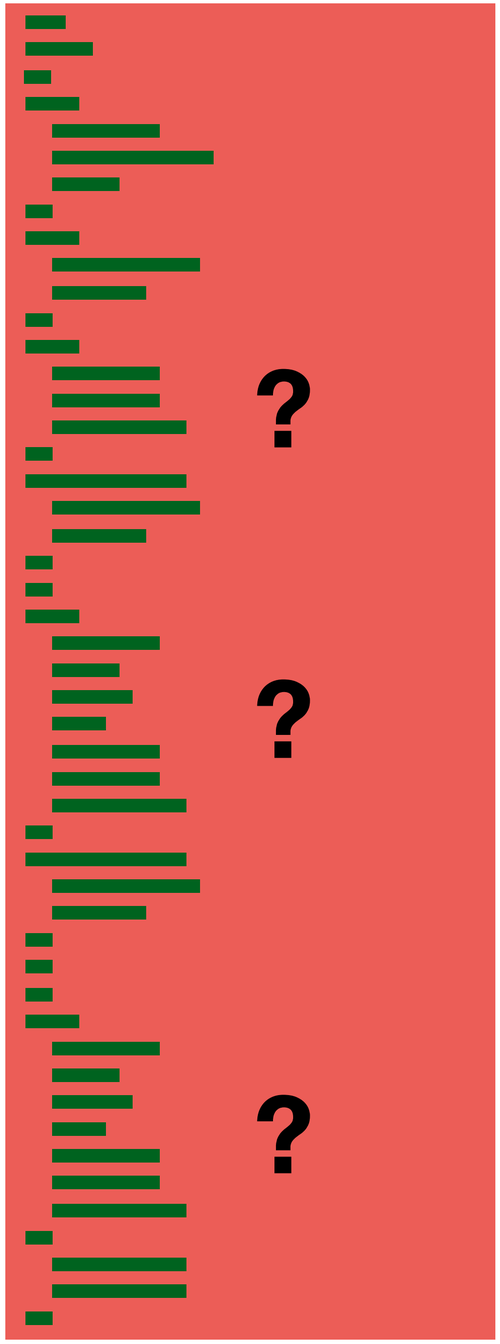
Agile Planning

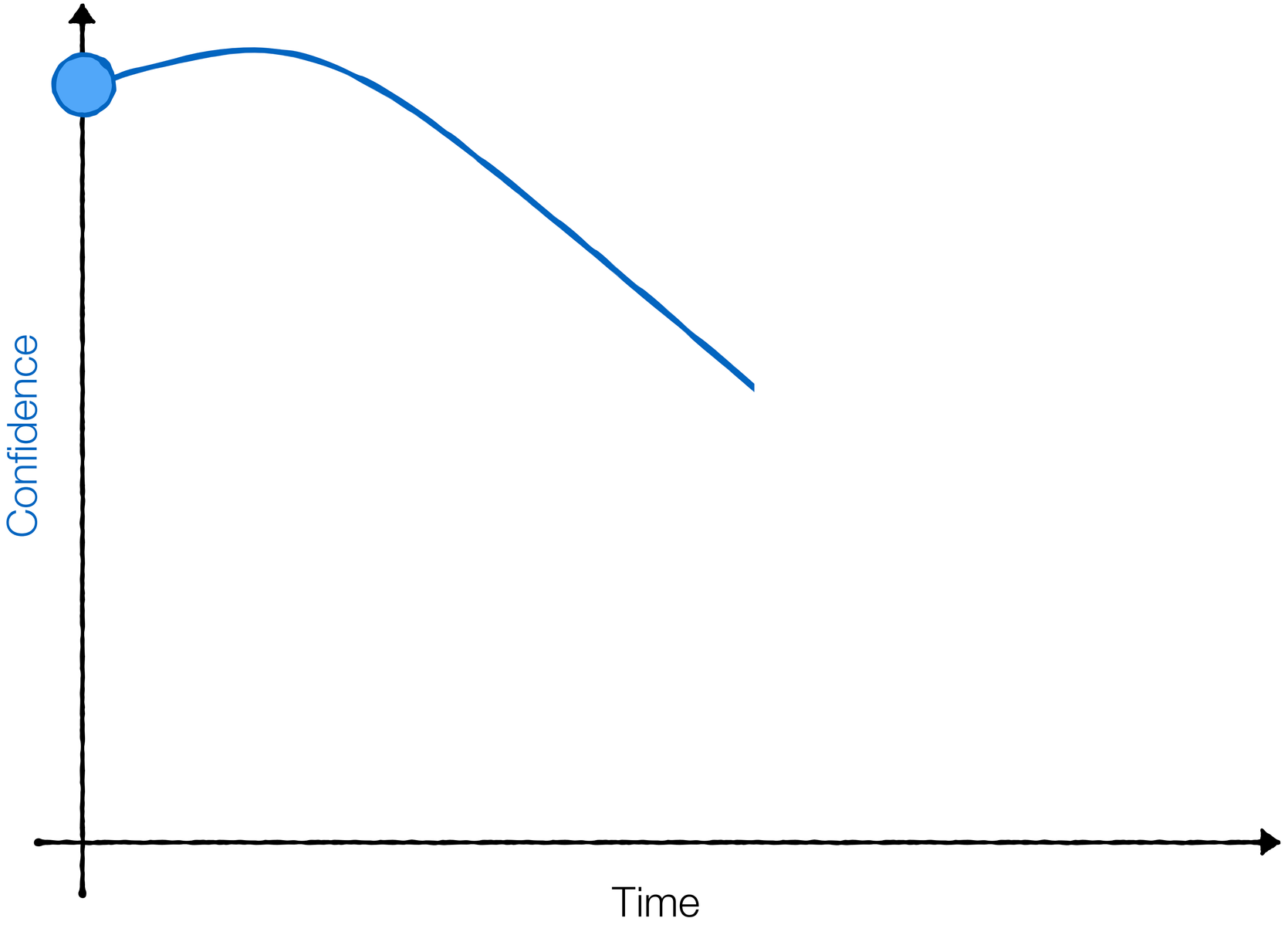
Confidence

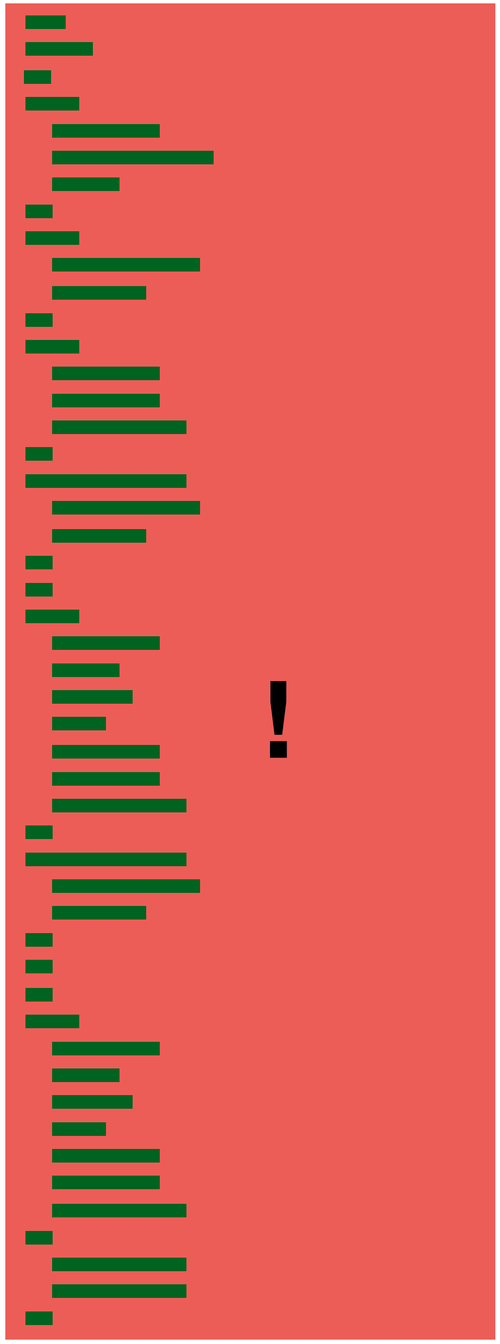

Confidence
Time

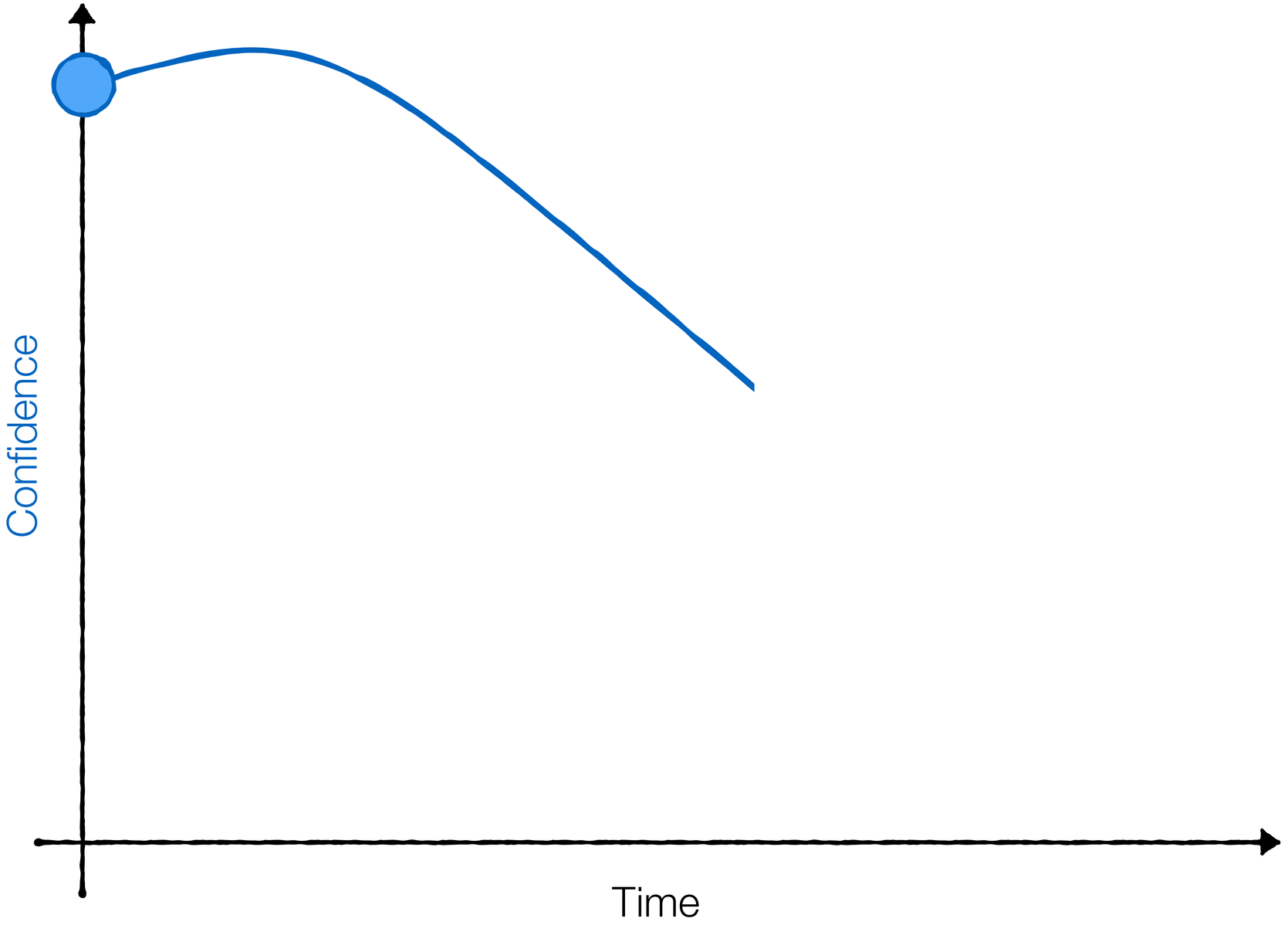

Confidence
Time

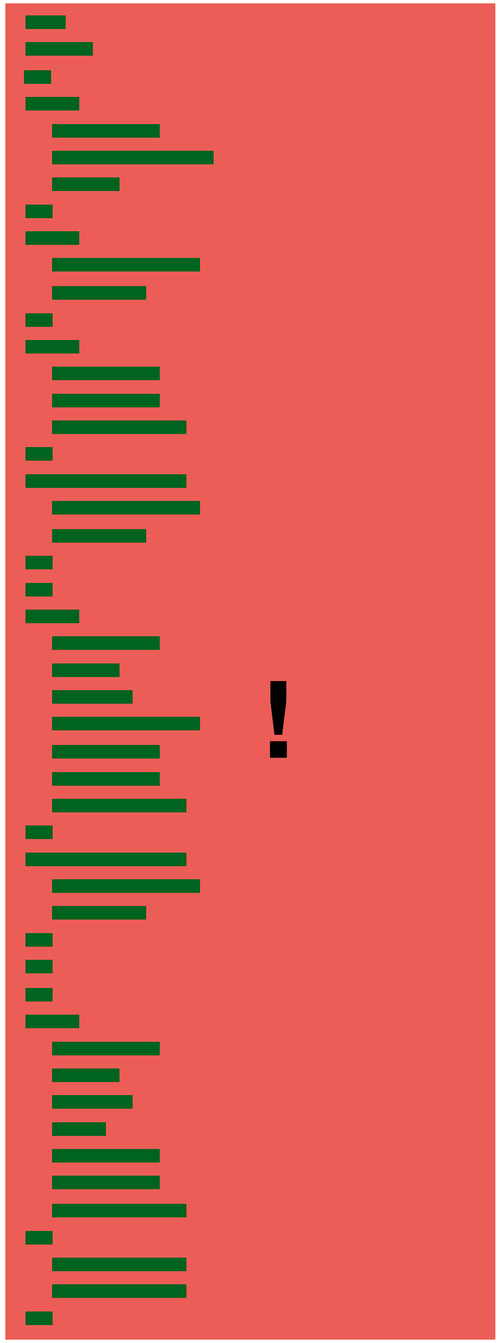

Confidence
Time

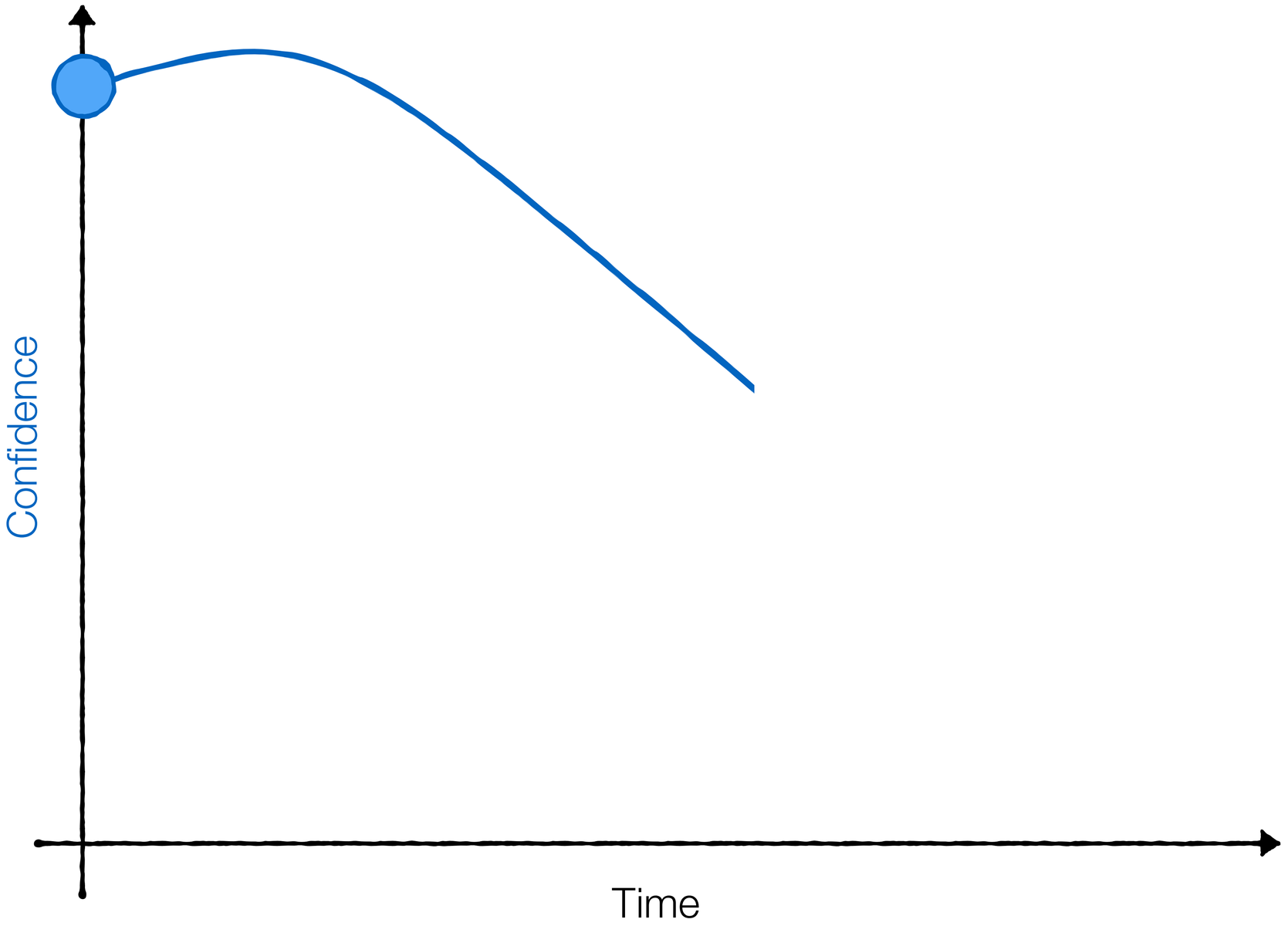

?
?
?

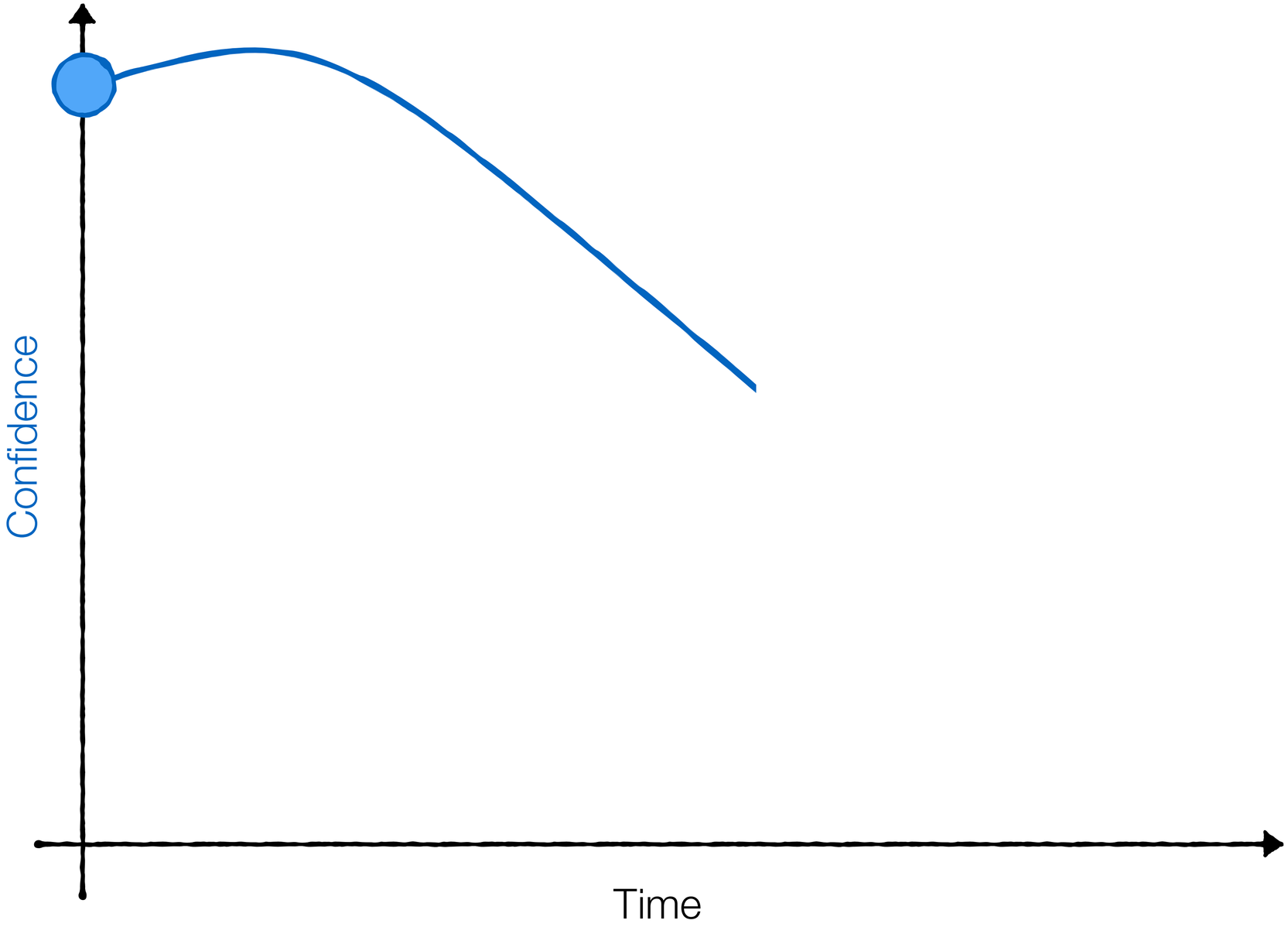

Confidence
Time

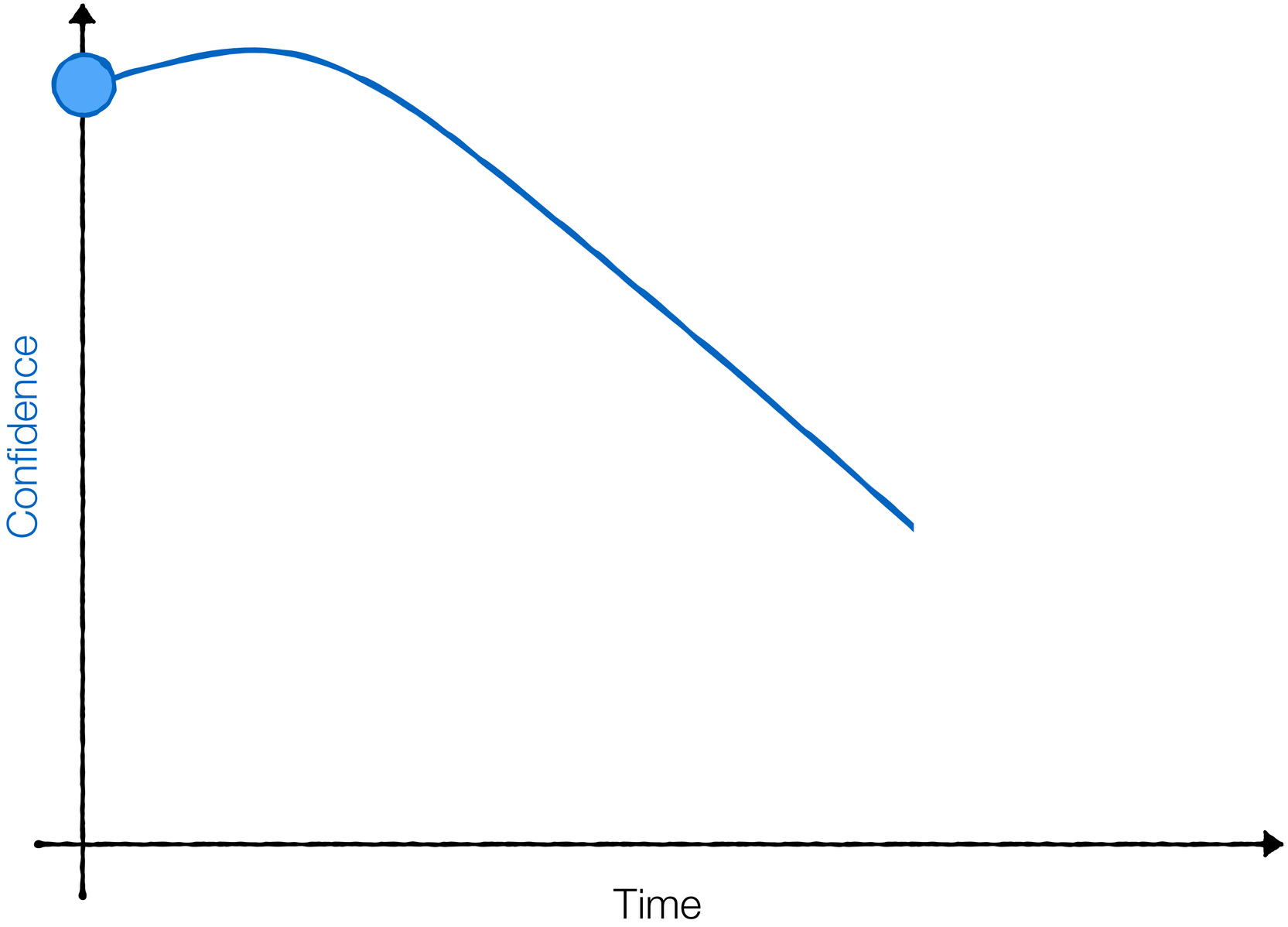

!

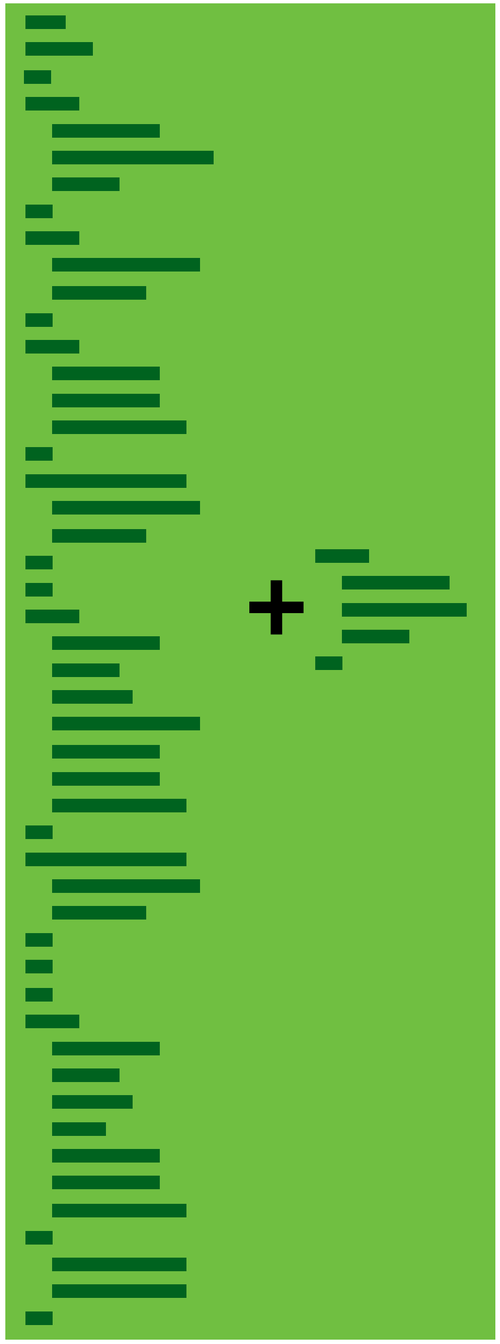

Confidence
Time

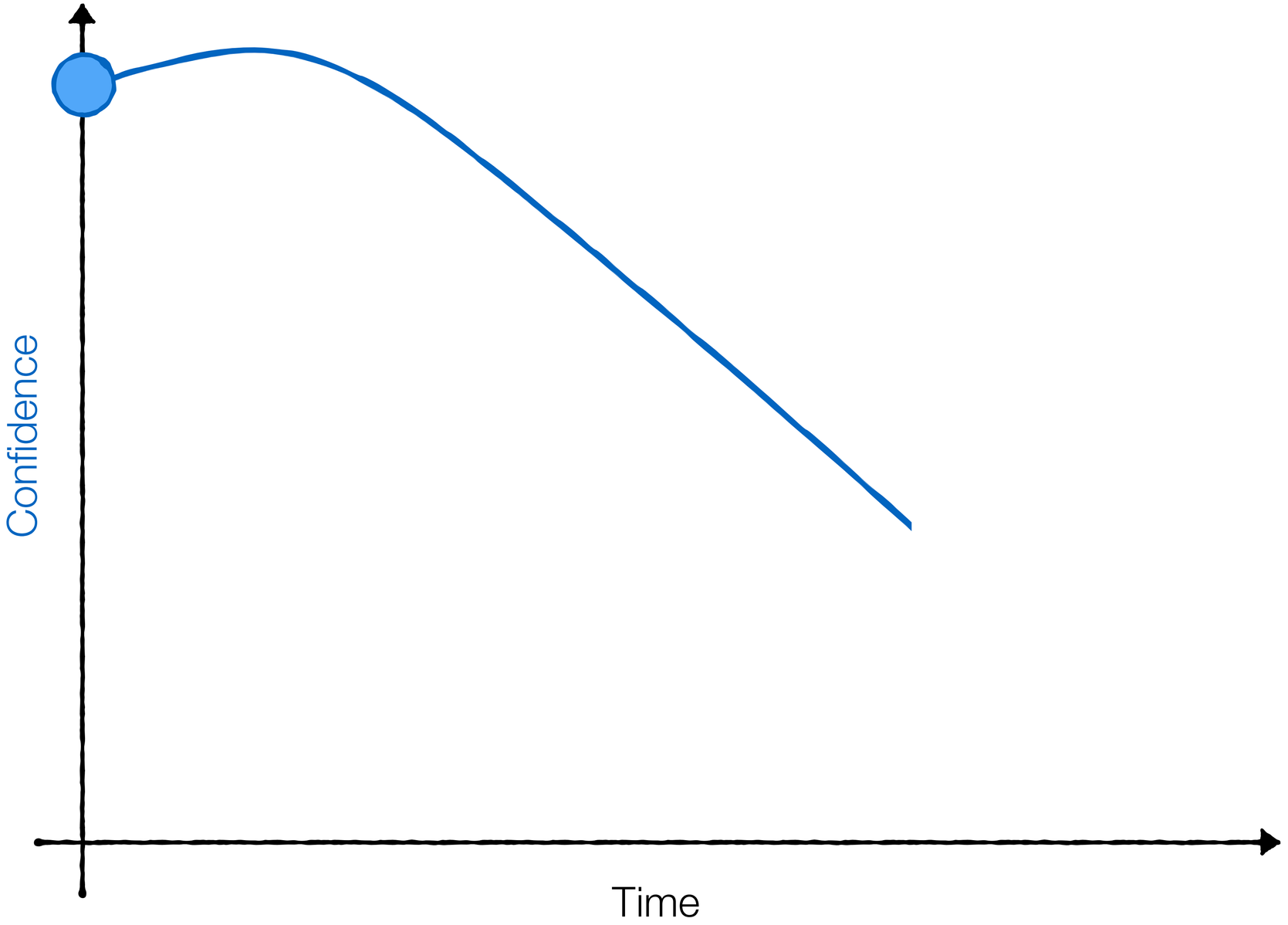

!

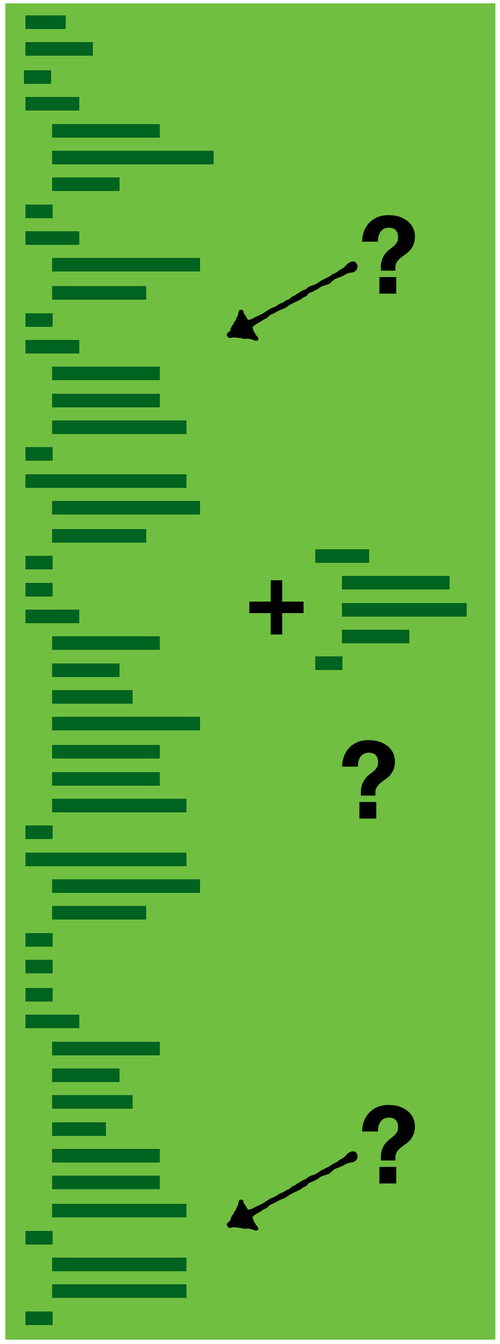

Confidence
Time

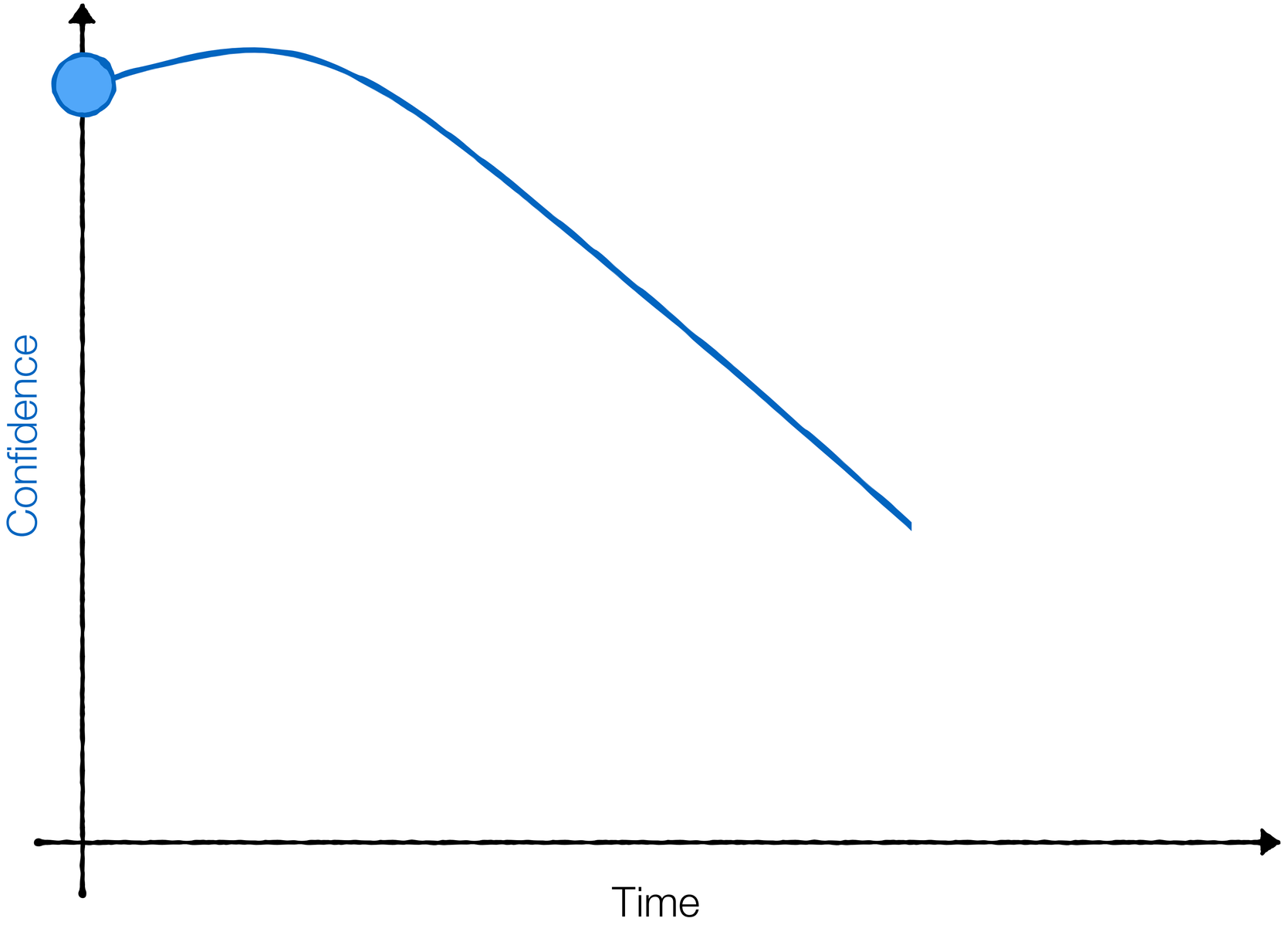

Confidence
Time

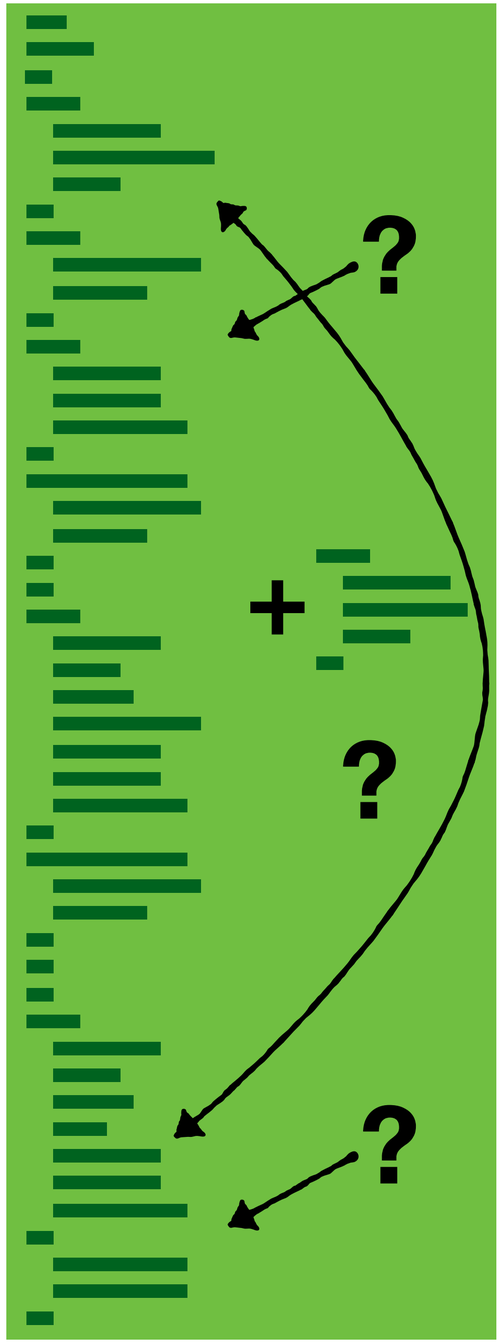

Confidence
Time

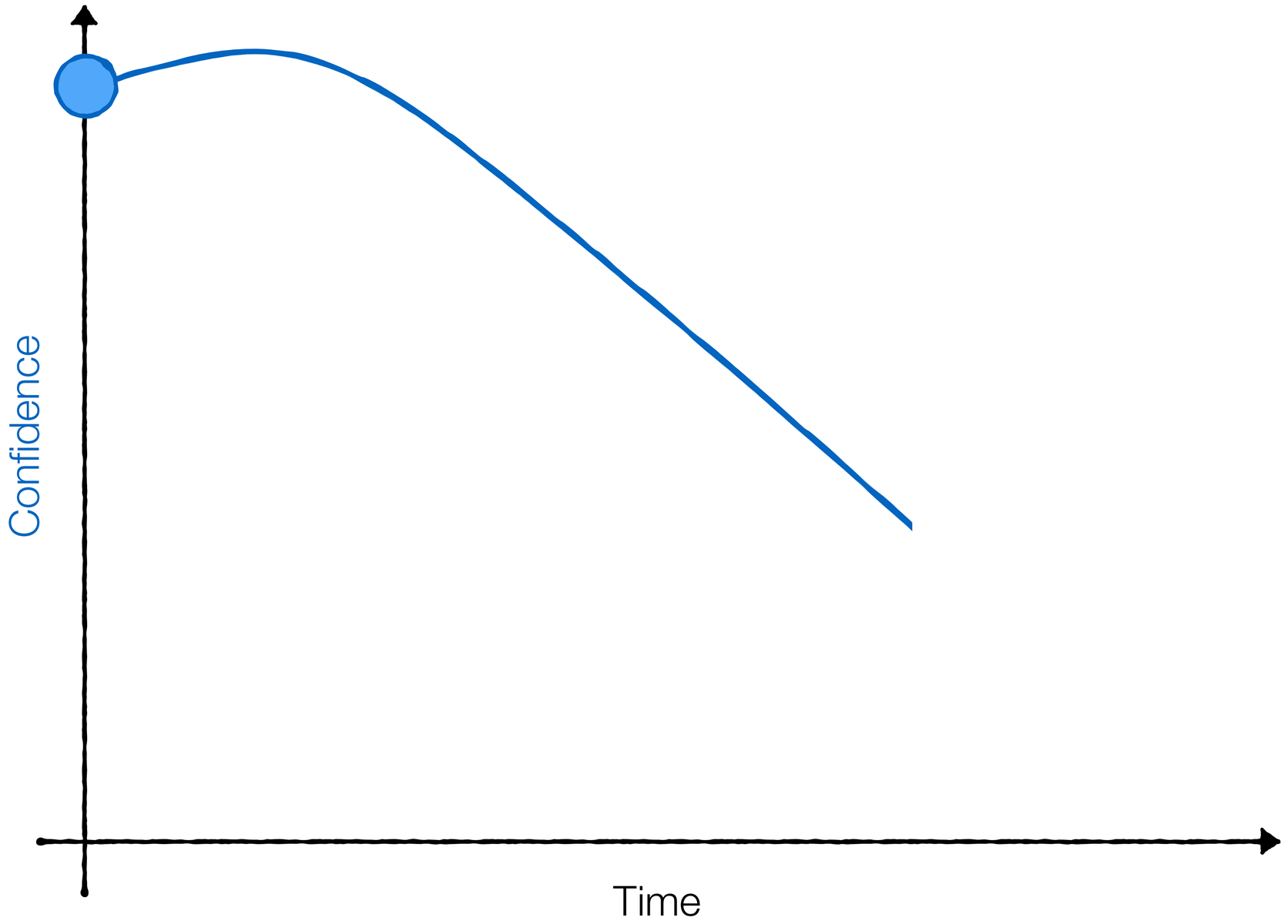


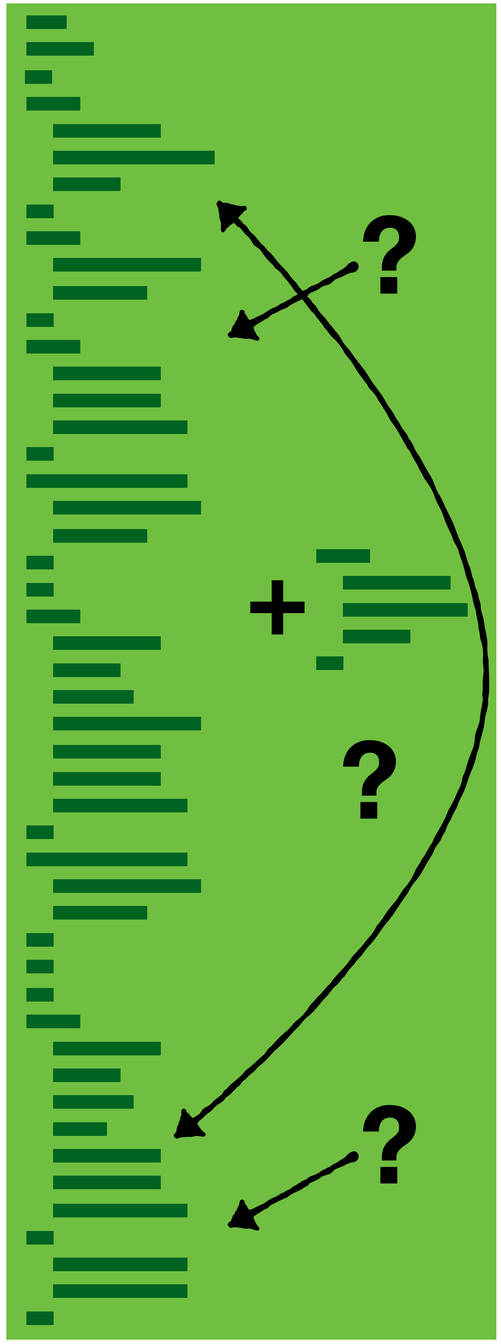

Confidence
Time

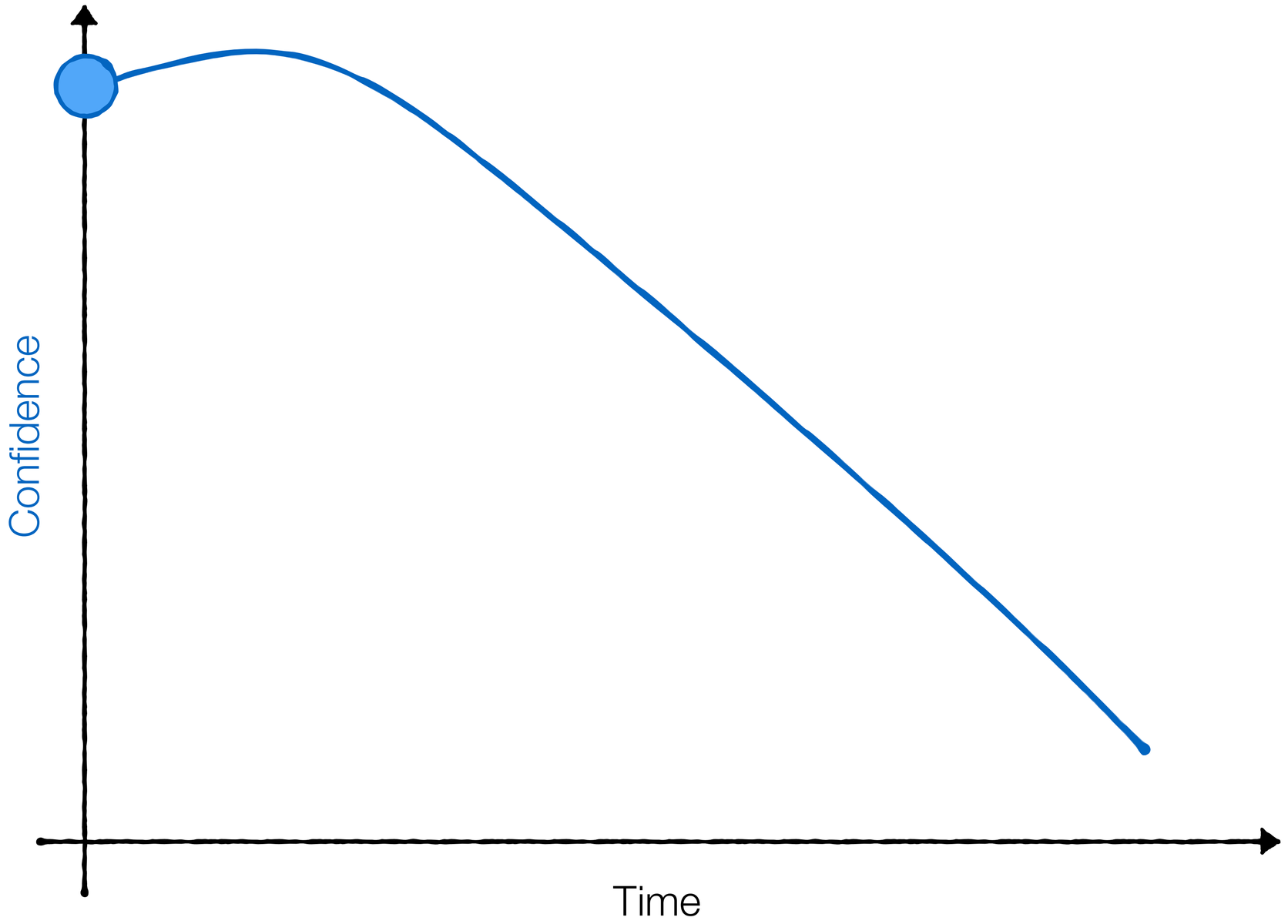

?
+
?
?

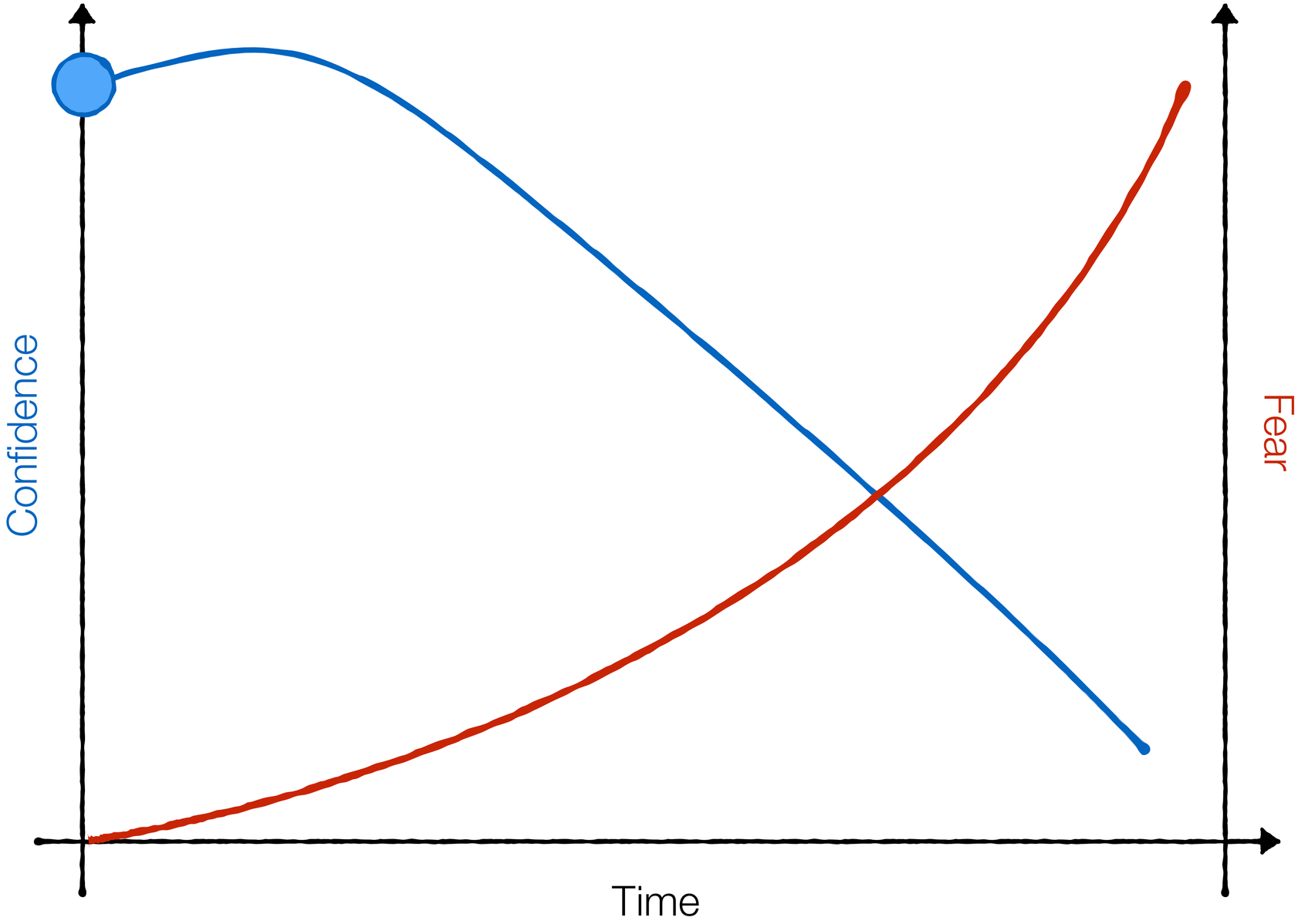

Confidence
Time


?
+
-
?
?


Confidence
Time

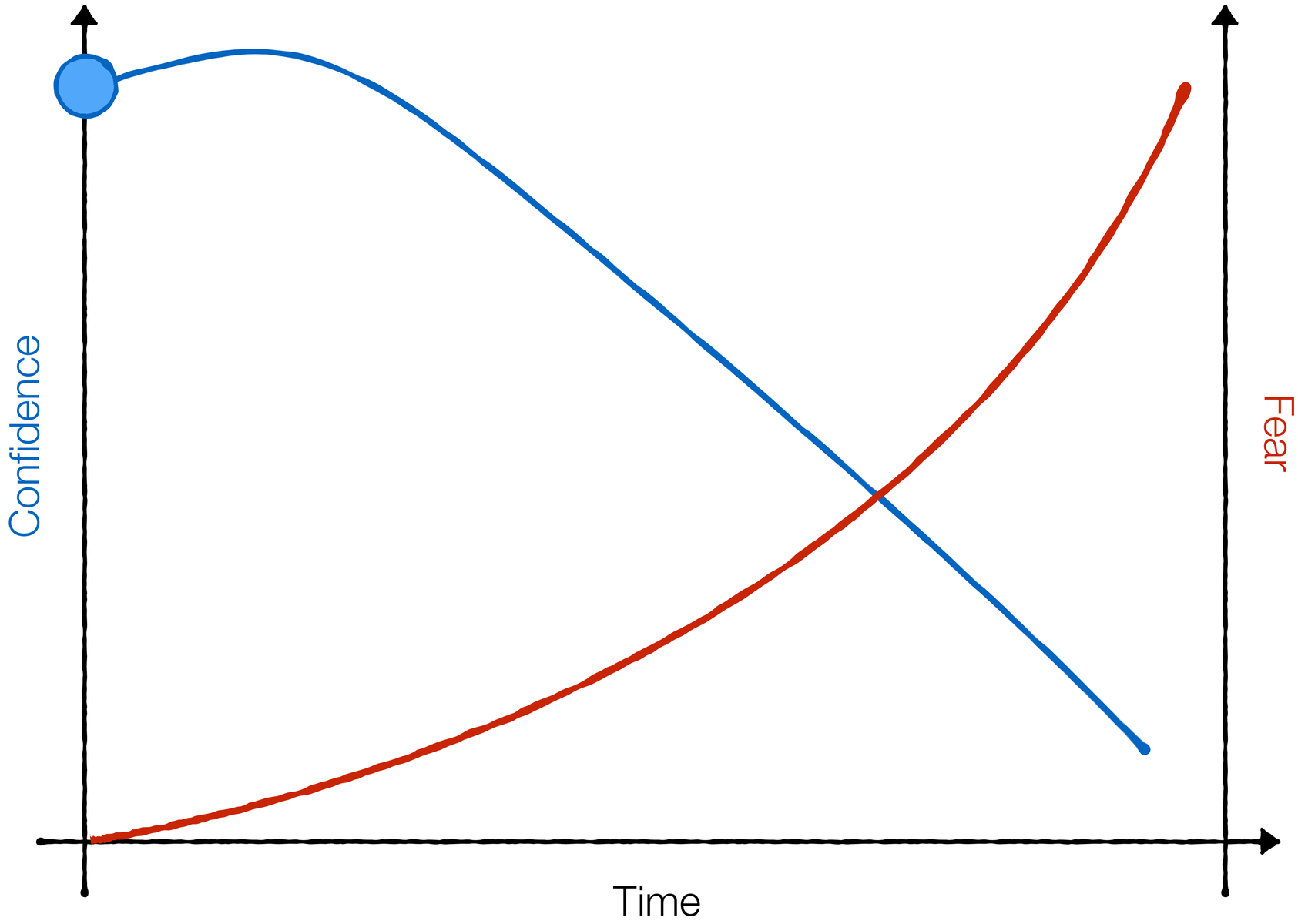

?
+
-
?
?


Confidence
Time


Confidence
Fear
Time

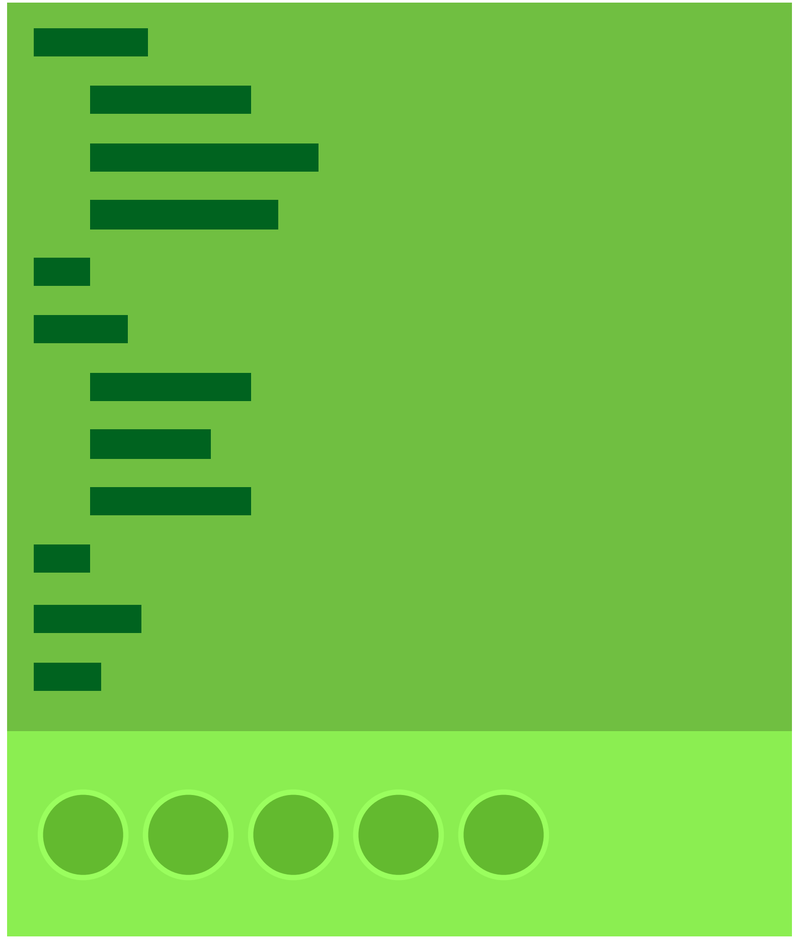
This

SUCKS

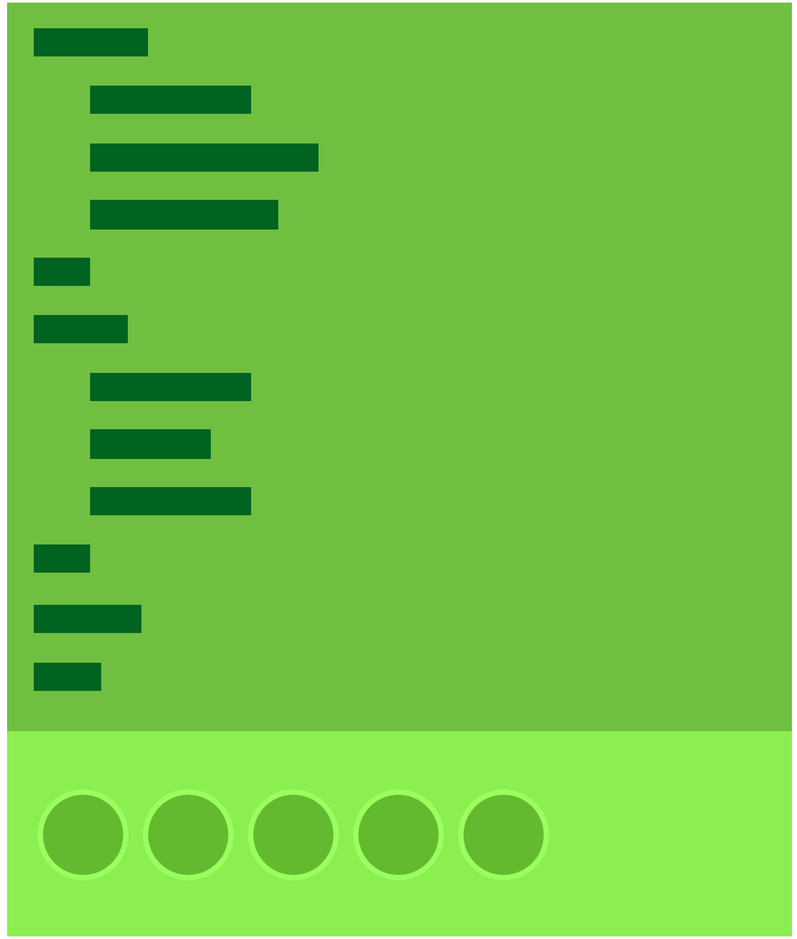


There is a better way…

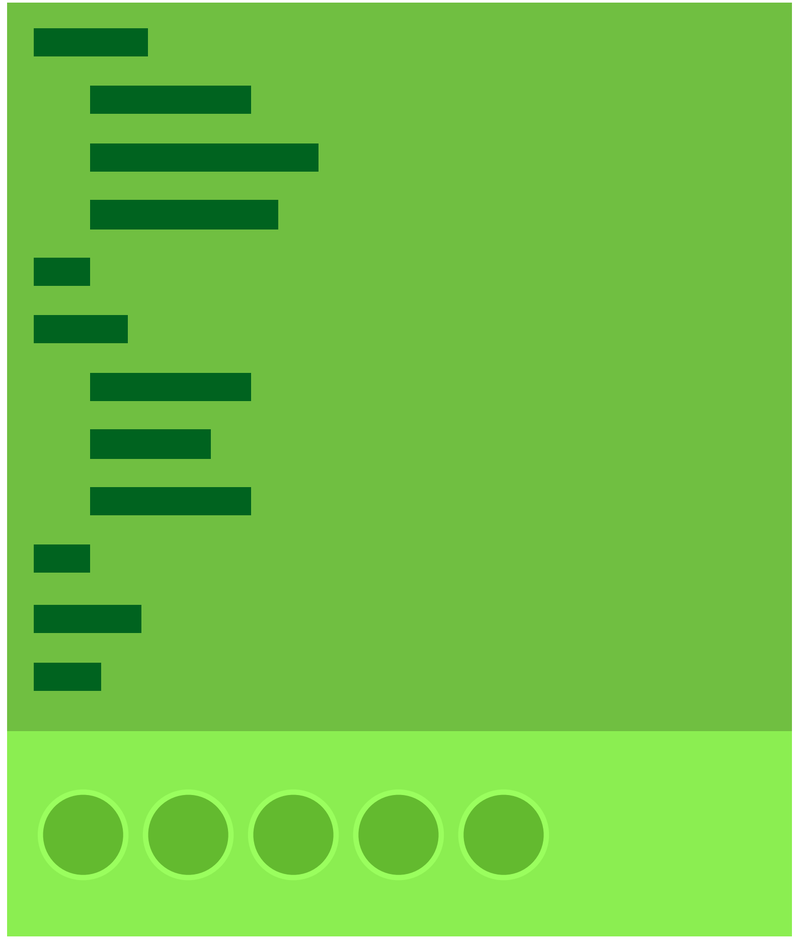
Automated tests!



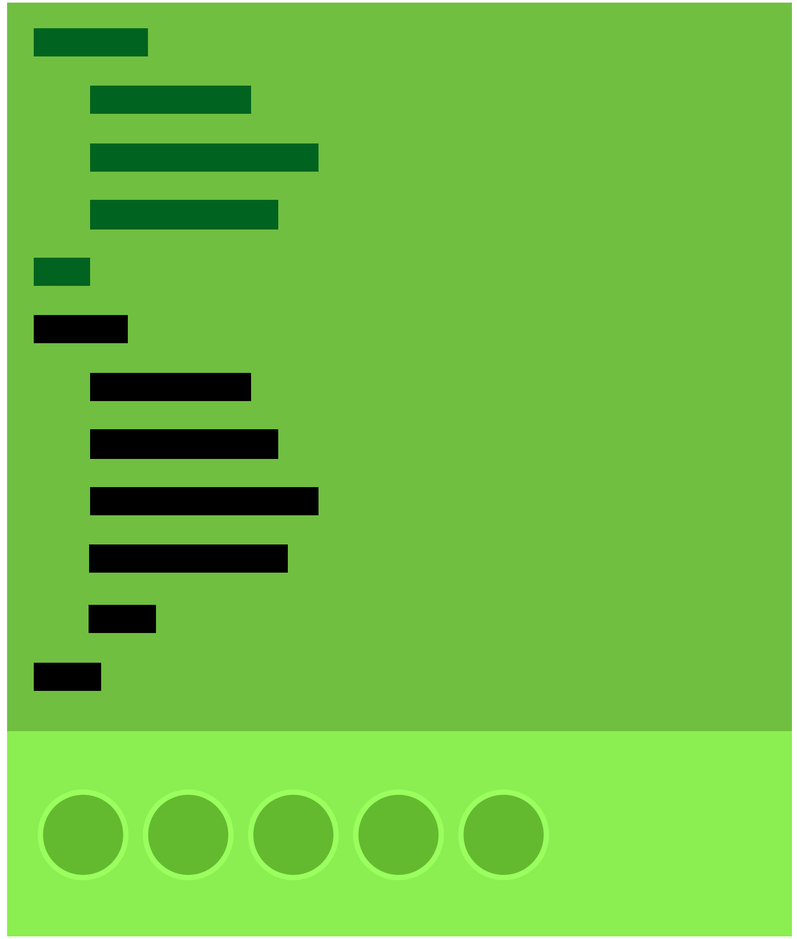
Automated tests!
Guarantee your code works



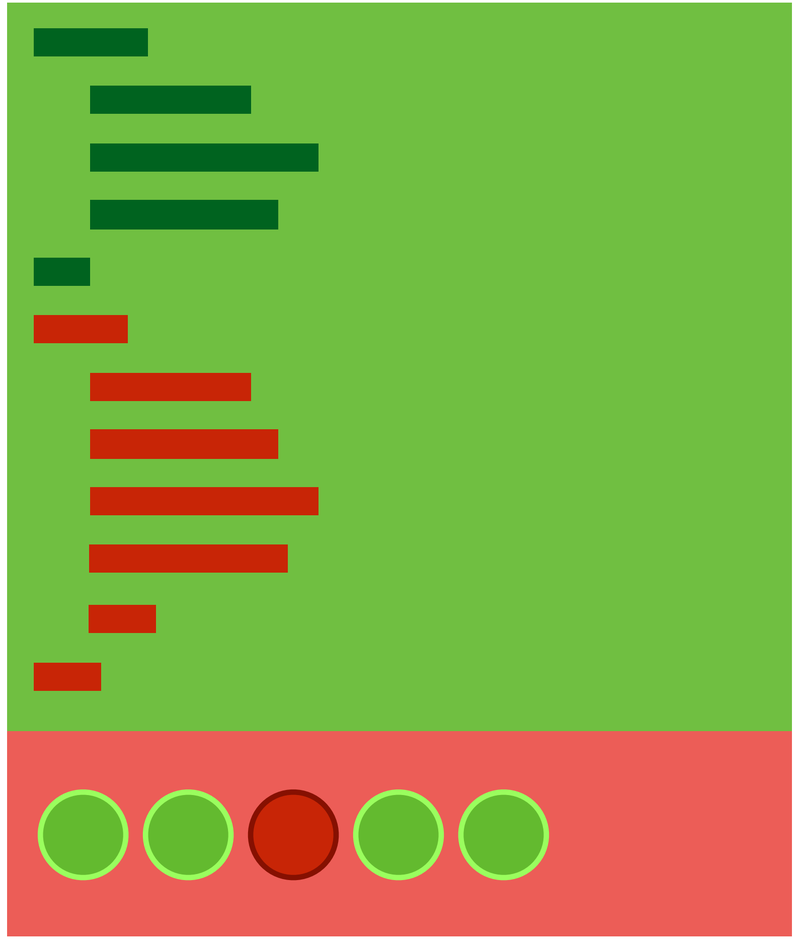
Automated tests!
Guarantee your code works
…as you refactor



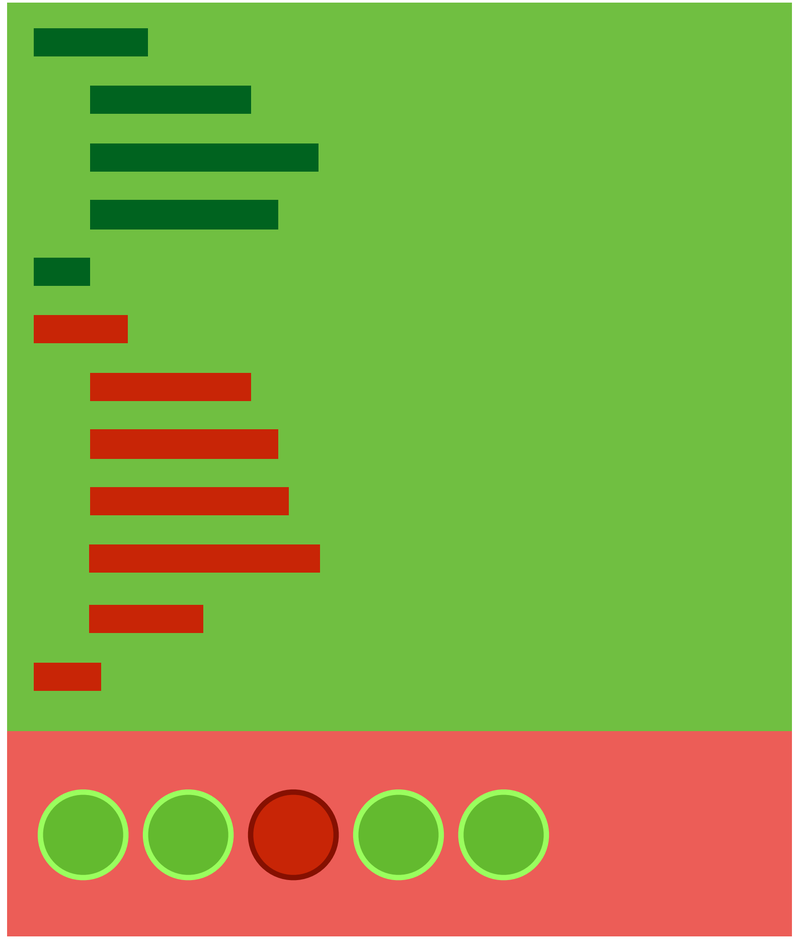
Automated tests!
Guarantee your code works
…as you refactor



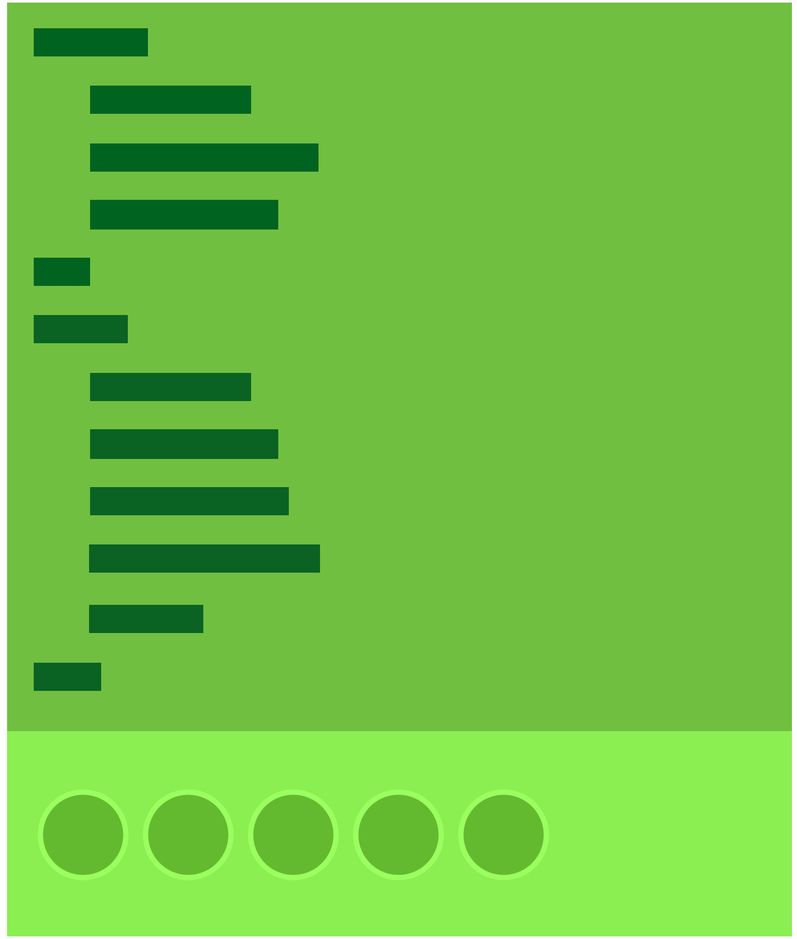
Automated tests!
Guarantee your code works
…as you refactor



Automated tests!
Guarantee your code works
…as you refactor



Automated tests!
Guarantee your code works
…as you refactor

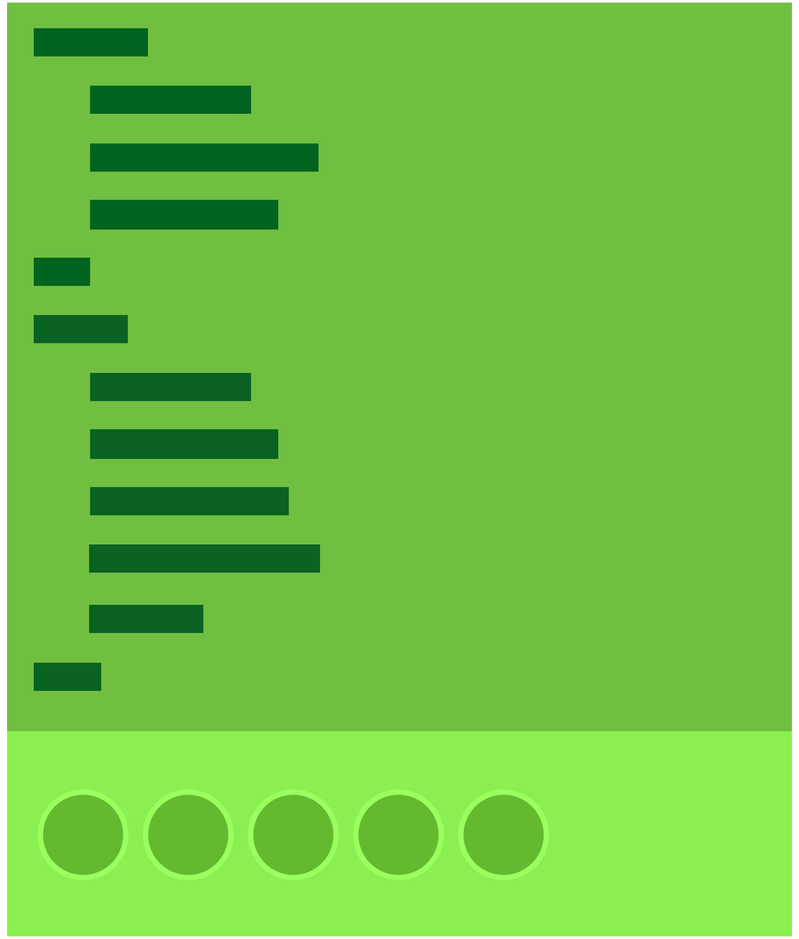


Automated tests!

Guarantee your code works

…as you refactor

…as you add new features

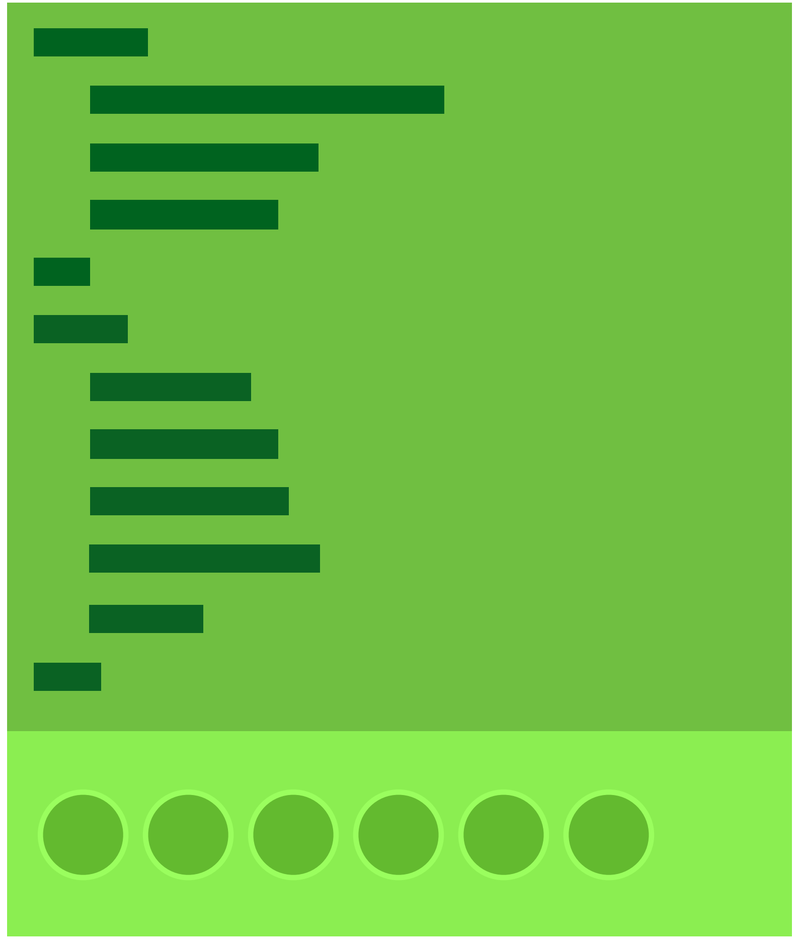


Automated tests!

Guarantee your code works

…as you refactor

…as you add new features



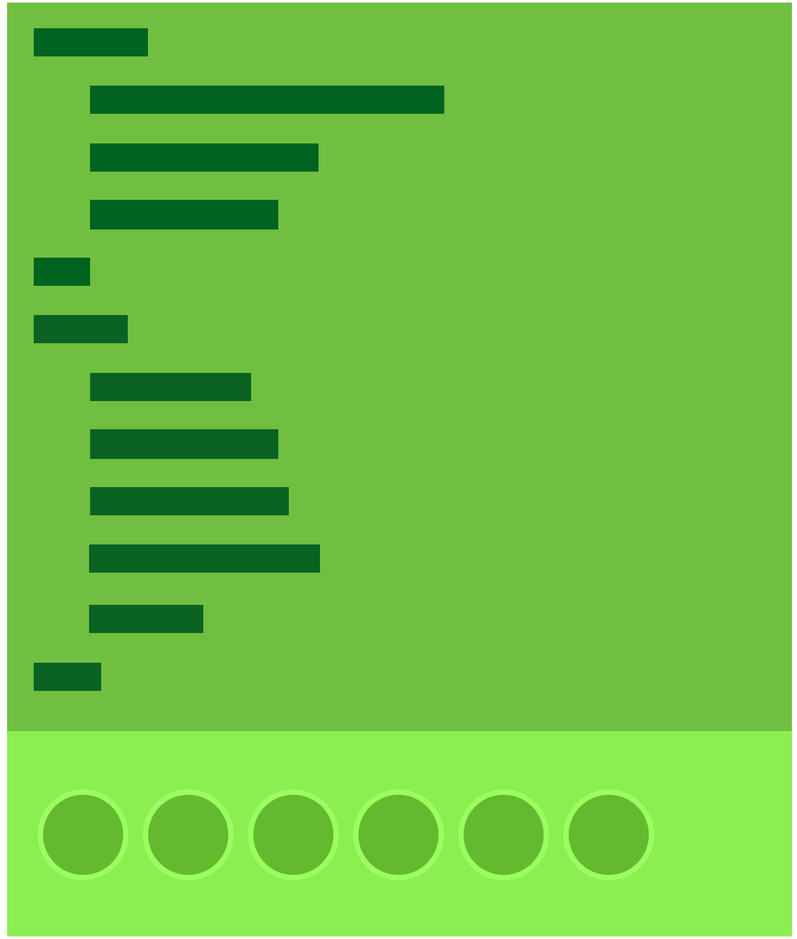
Automated tests!

Guarantee your code works

…as you refactor

…as you add new features

…as you fix bugs



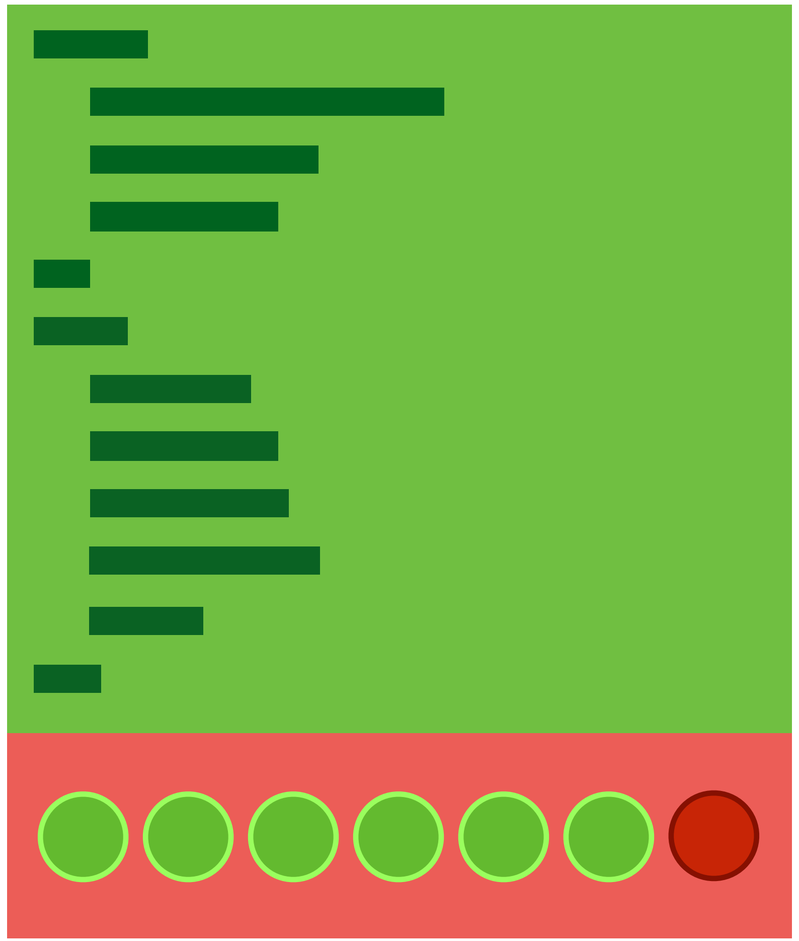
Automated tests!

Guarantee your code works

…as you refactor

…as you add new features

…as you fix bugs



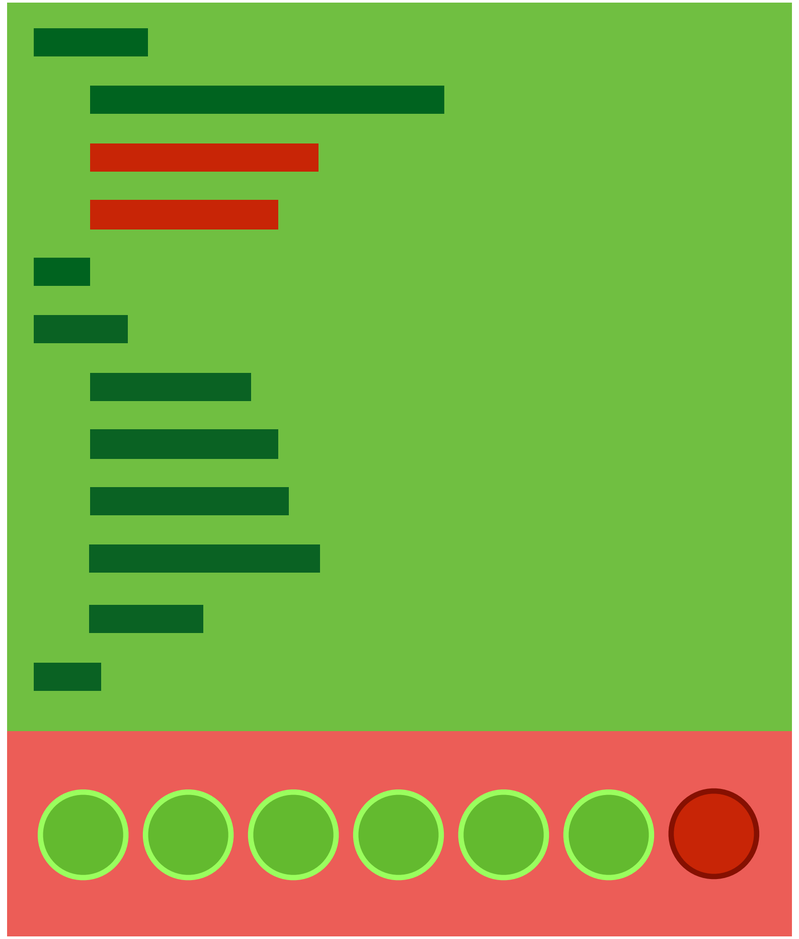
Automated tests!

Guarantee your code works

…as you refactor

…as you add new features

…as you fix bugs



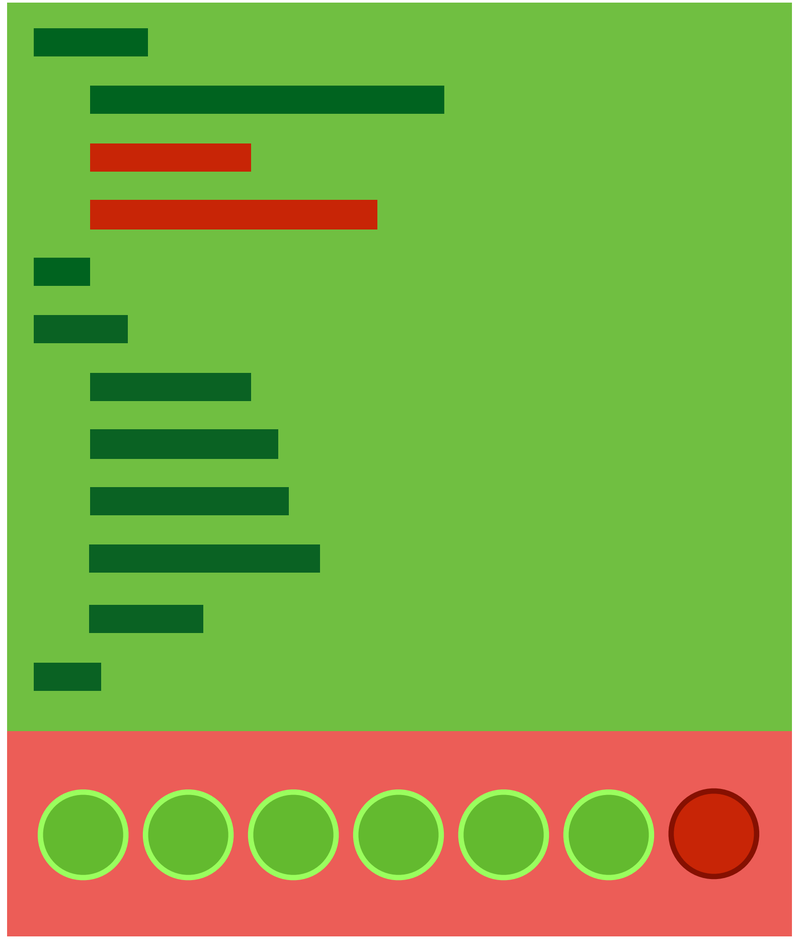
Automated tests!

Guarantee your code works

…as you refactor

…as you add new features

…as you fix bugs



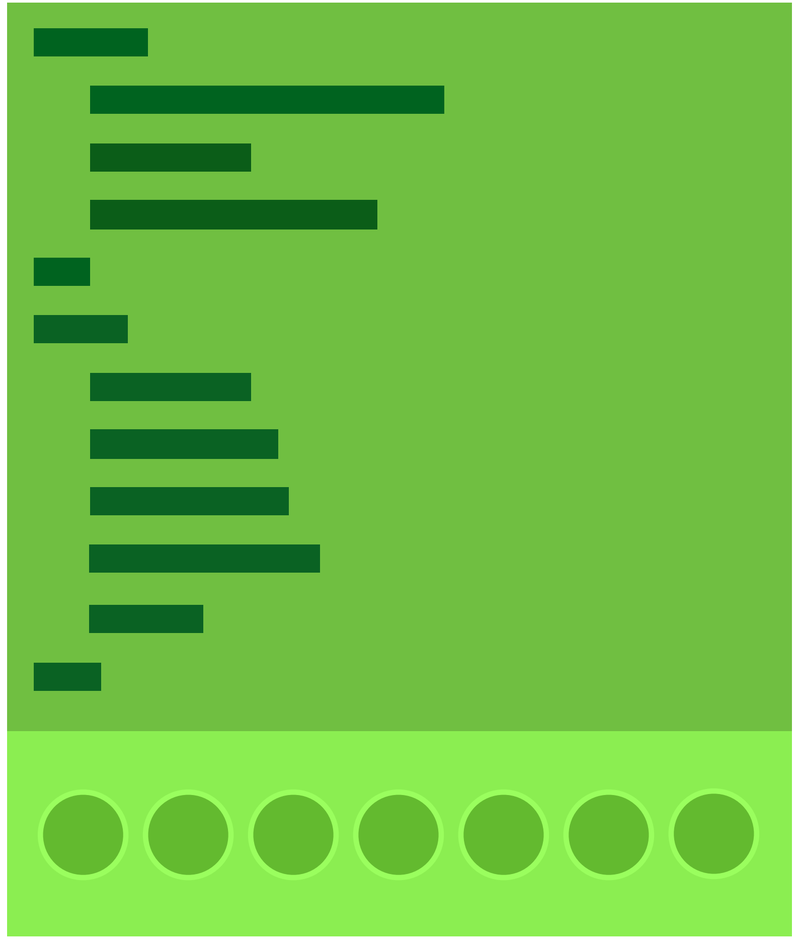
Automated tests!

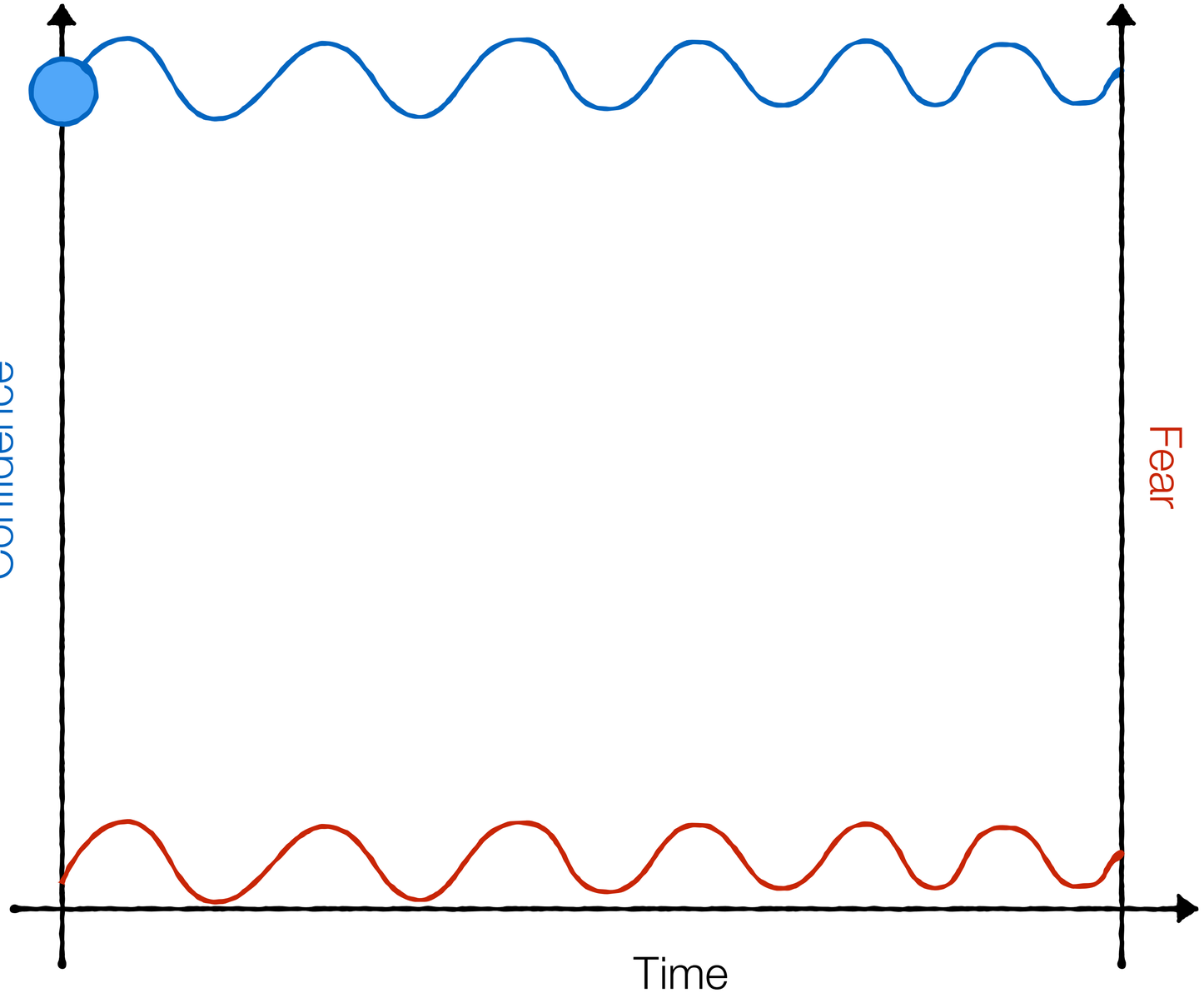
Guarantee your code works

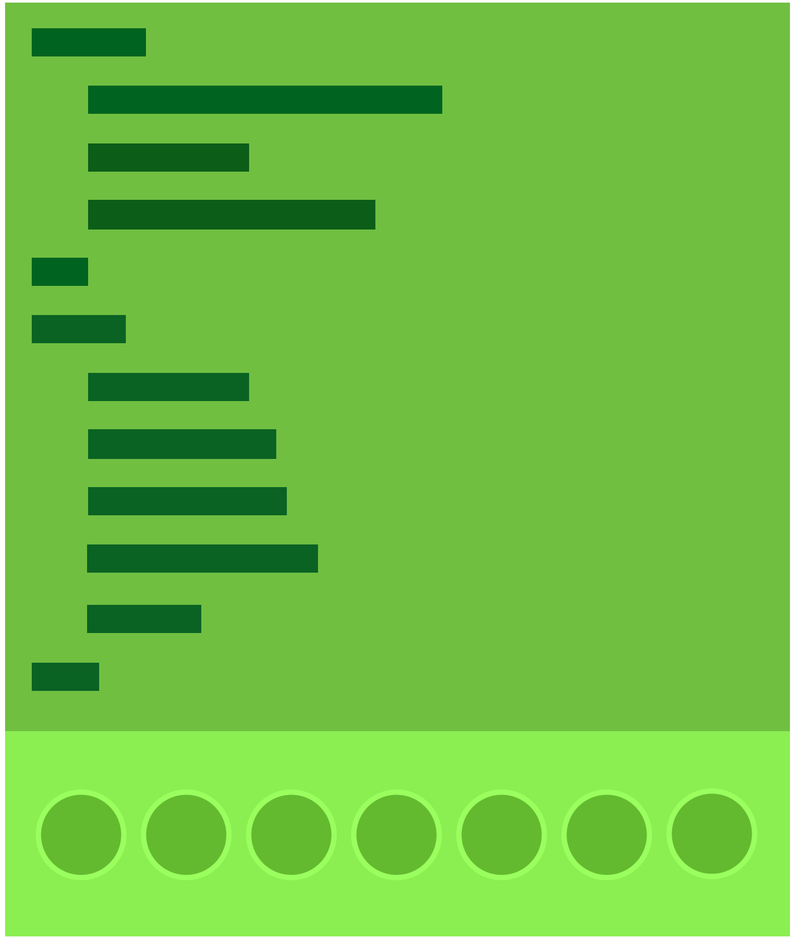
…as you refactor

…as you add new features

…as you fix bugs




Confidence
Fear
Time



Testing


Pivotal
TM

TDD: Test-Driven-Development

TDD: Test-Driven-Development

=

Testing

TDD: Test-Driven-Development

=

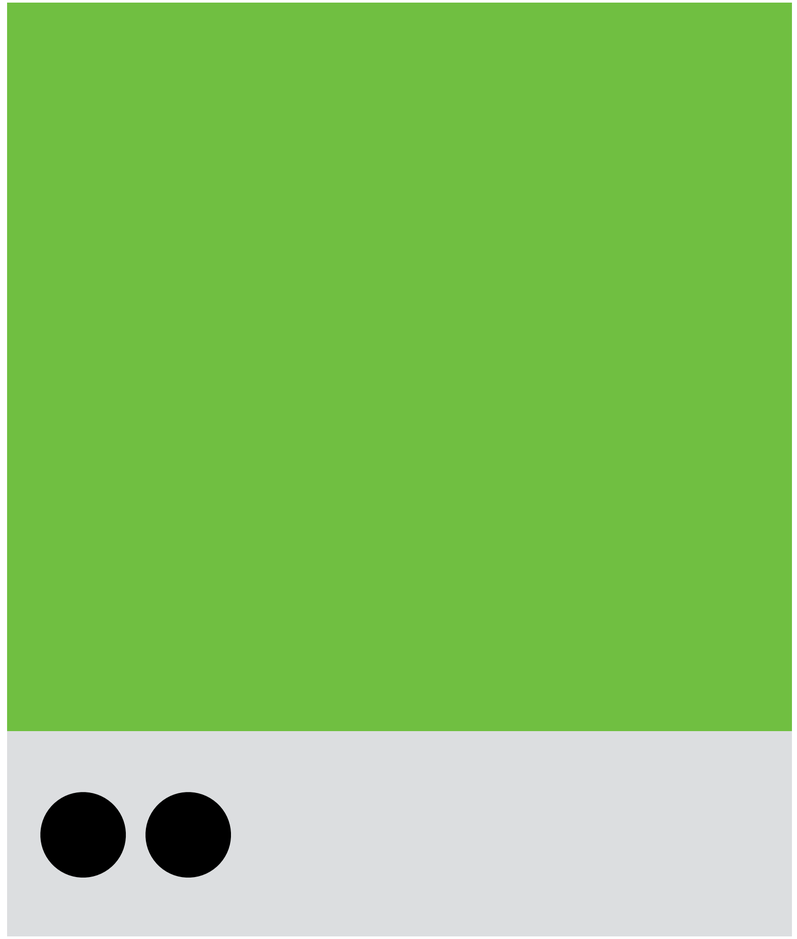
Testing

+

Discipline

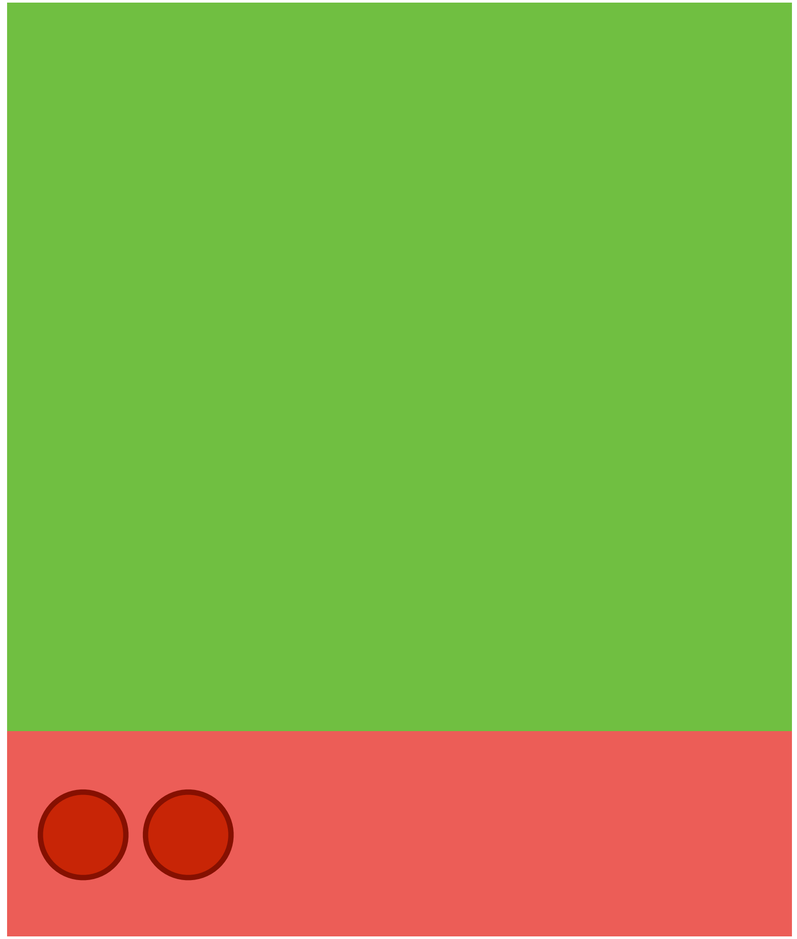
TDD Cycle

Tests



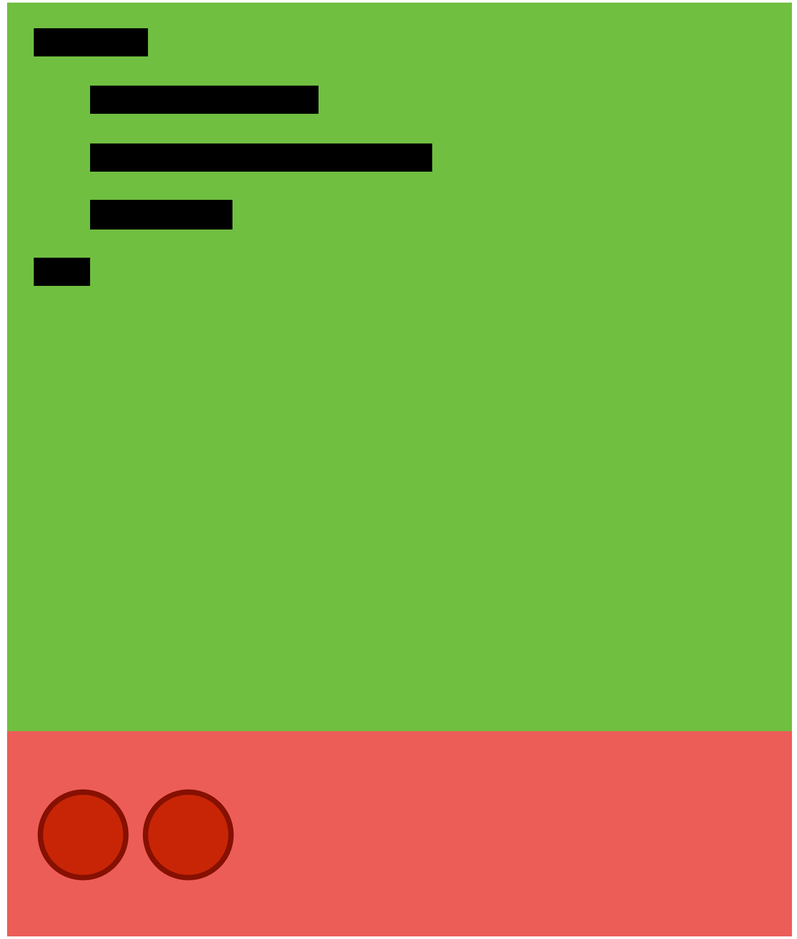
TDD Cycle

Tests, red



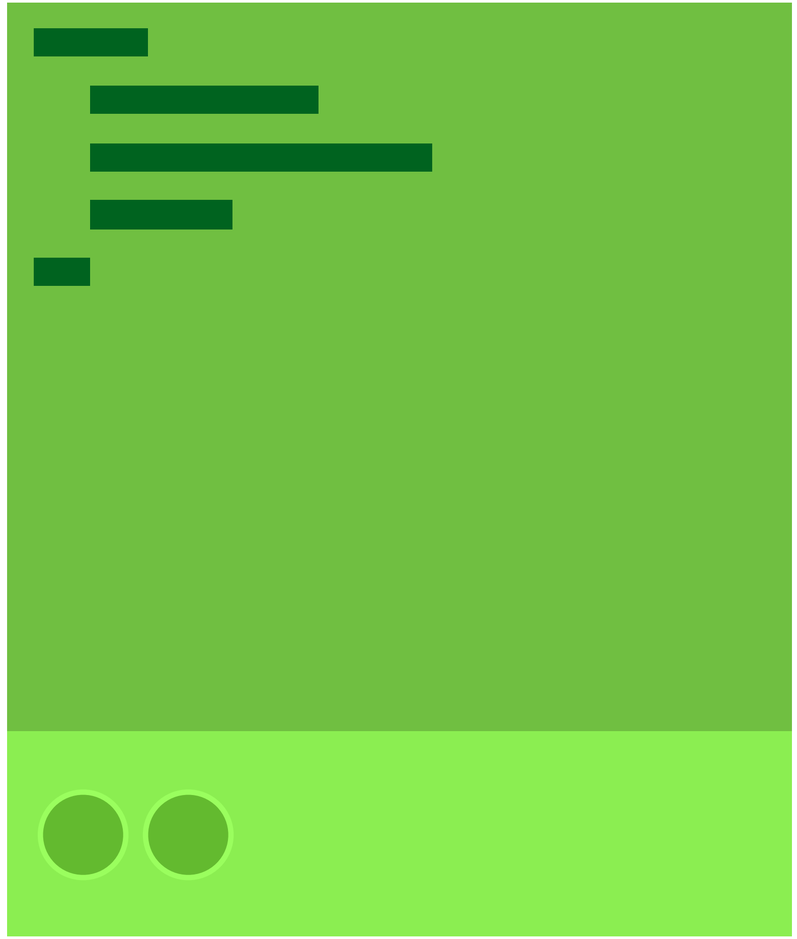
TDD Cycle

Tests, red, code



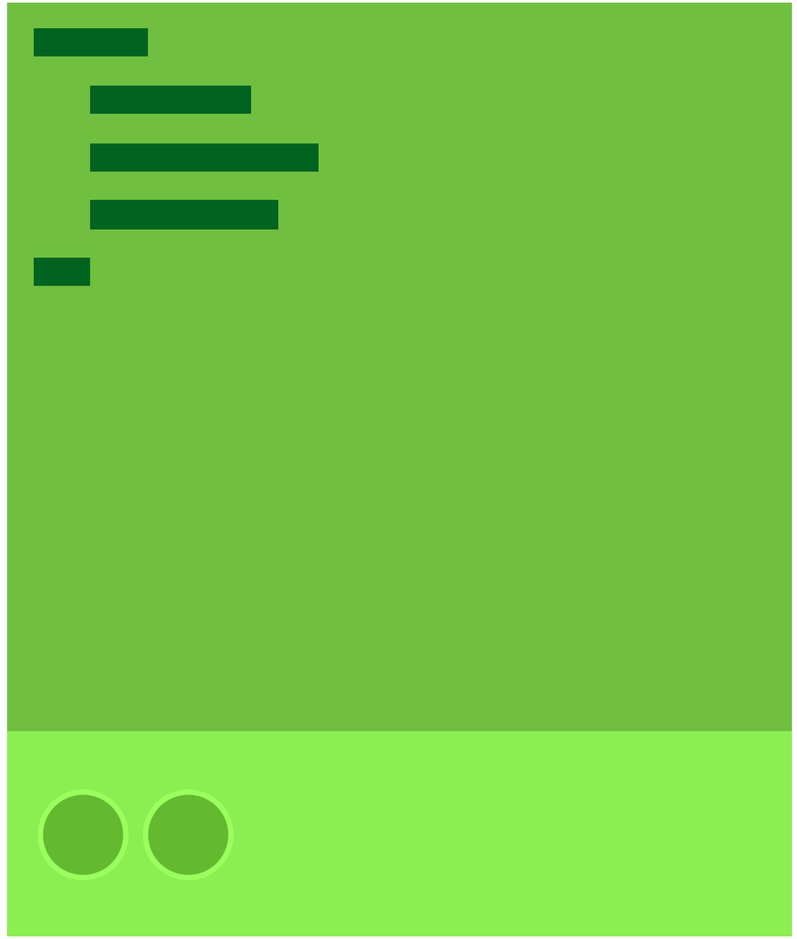
TDD Cycle

Tests, red, code, green



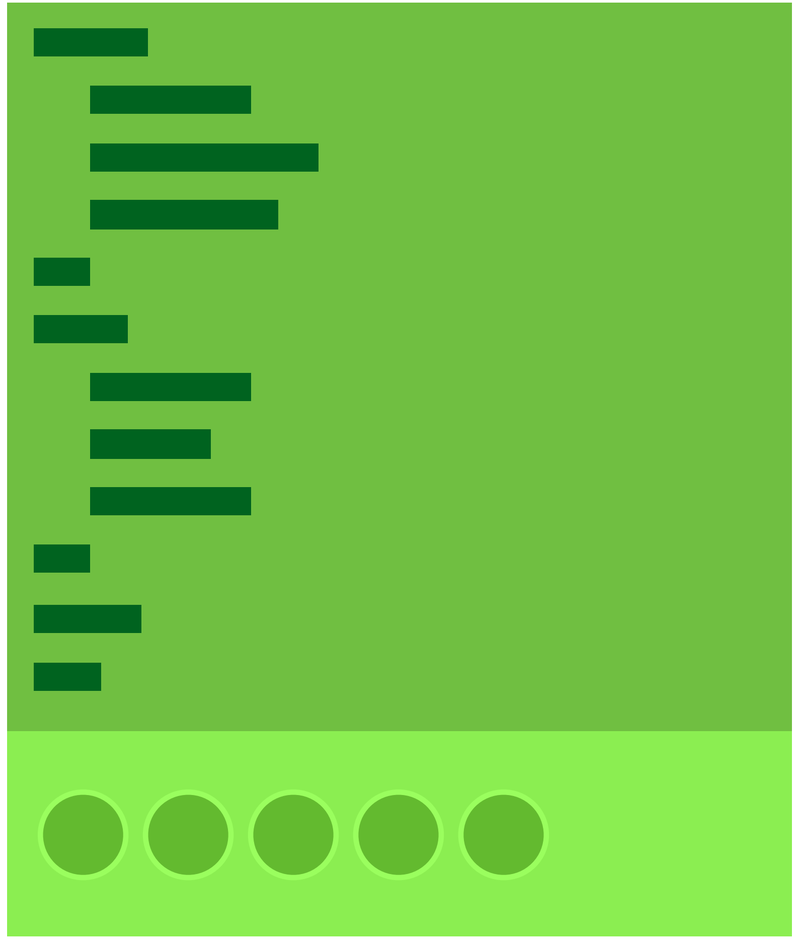
TDD Cycle

Tests, red, code, green, refactor



TDD Cycle

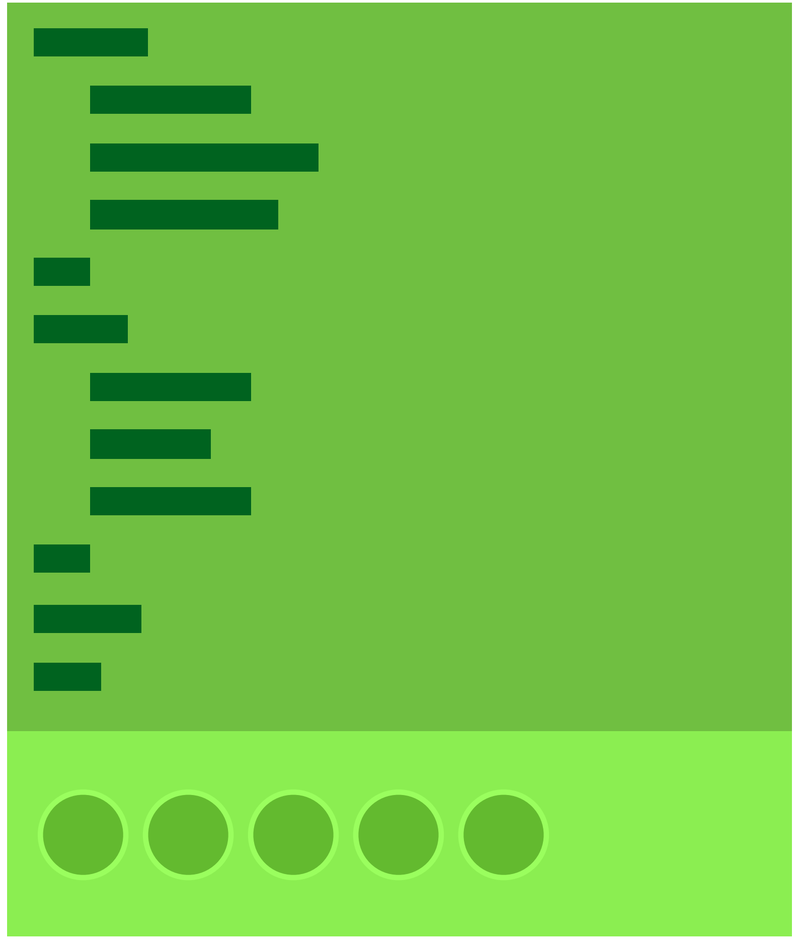
Tests, red, code, green, refactor, repeat



TDD Cycle

Tests, red, code, green, refactor, repeat

Why?

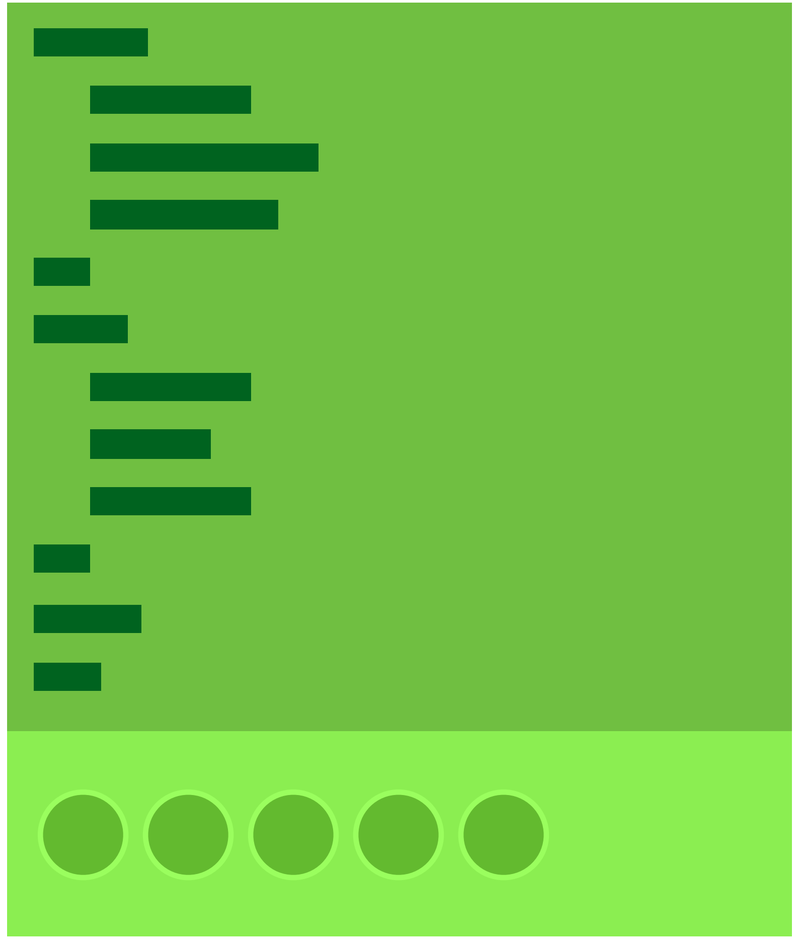


TDD Cycle

Tests, red, code, green, refactor, repeat

Why?

Test Quality



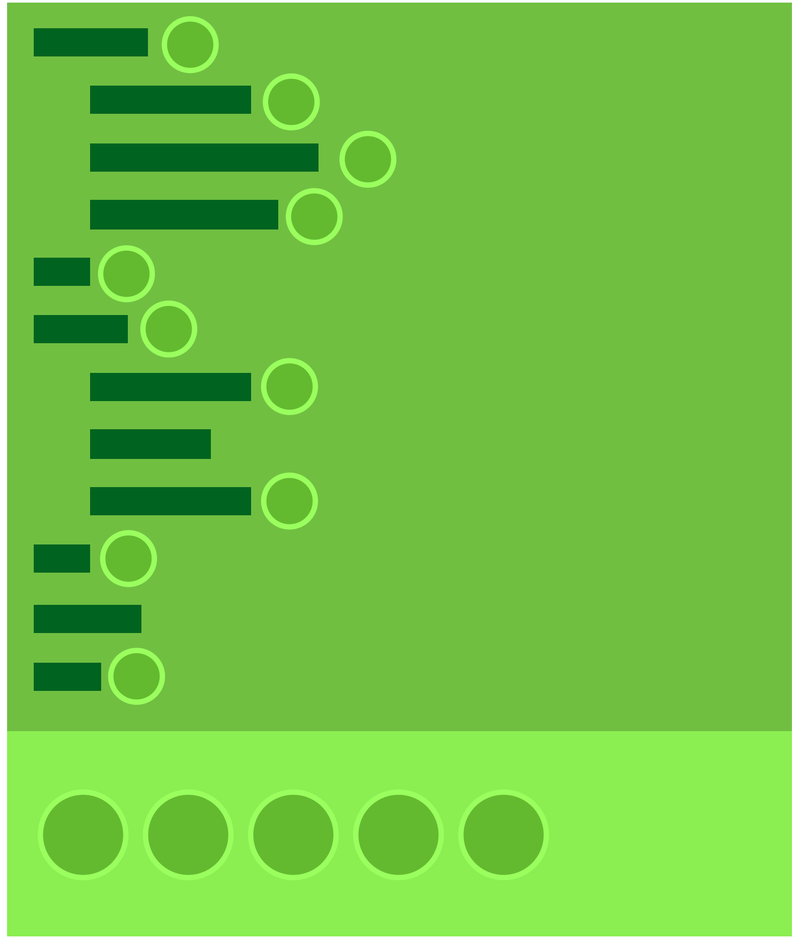
TDD Cycle

Tests, red, code, green, refactor, repeat

Why?

Test Quality

Coverage



TDD Cycle

Tests, red, code, green, refactor, repeat

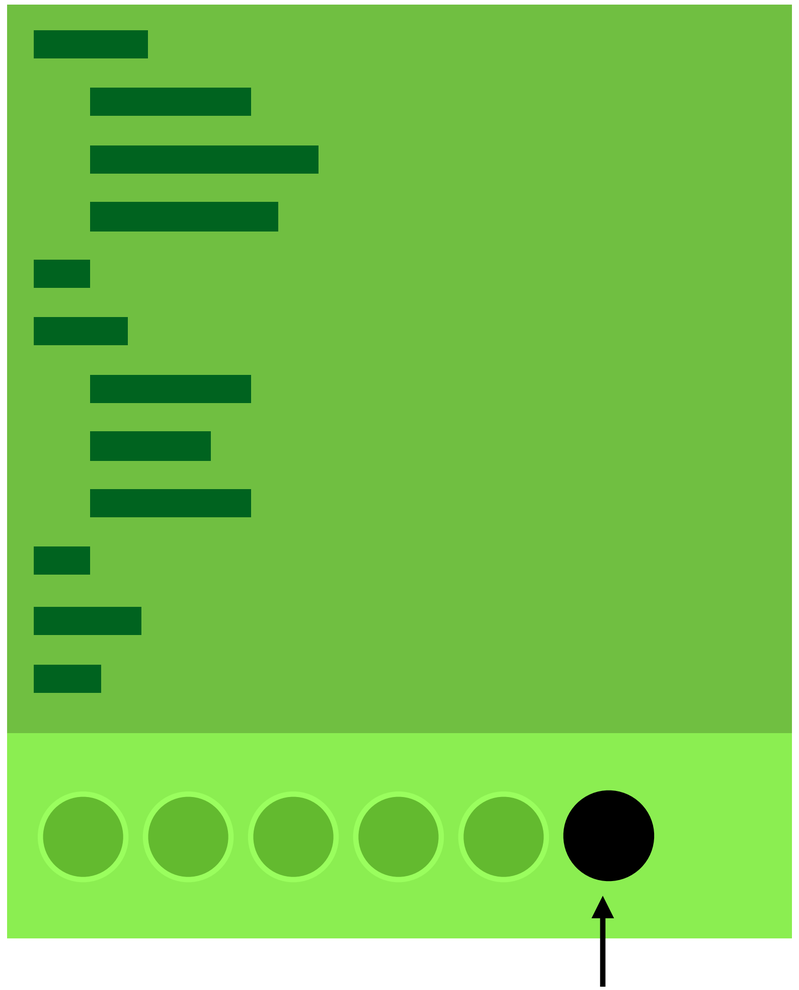
Why?

Test Quality

Coverage

Tests = Communication

1) Great when pair programming



TDD Cycle

Tests, red, code, green, refactor, repeat

Why?

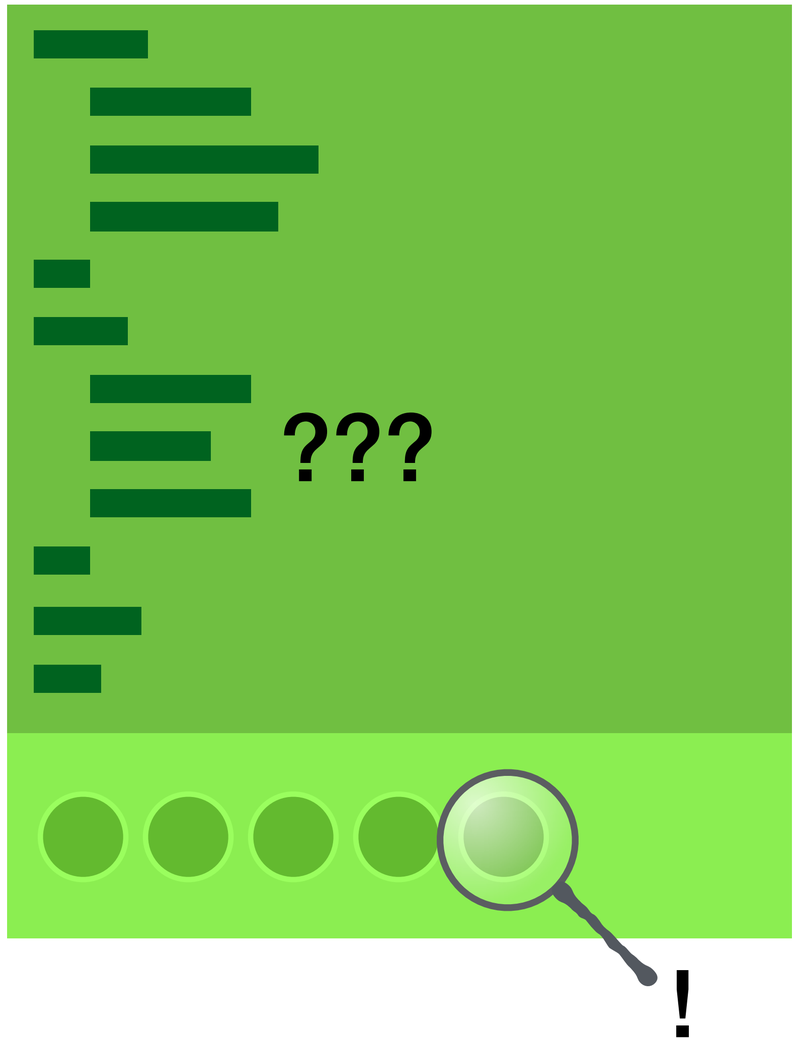
Test Quality

Coverage

Tests = Communication

1) Great when pair programming

2) Better documentation than comments



TDD Cycle

Tests, red, code, green, refactor, repeat

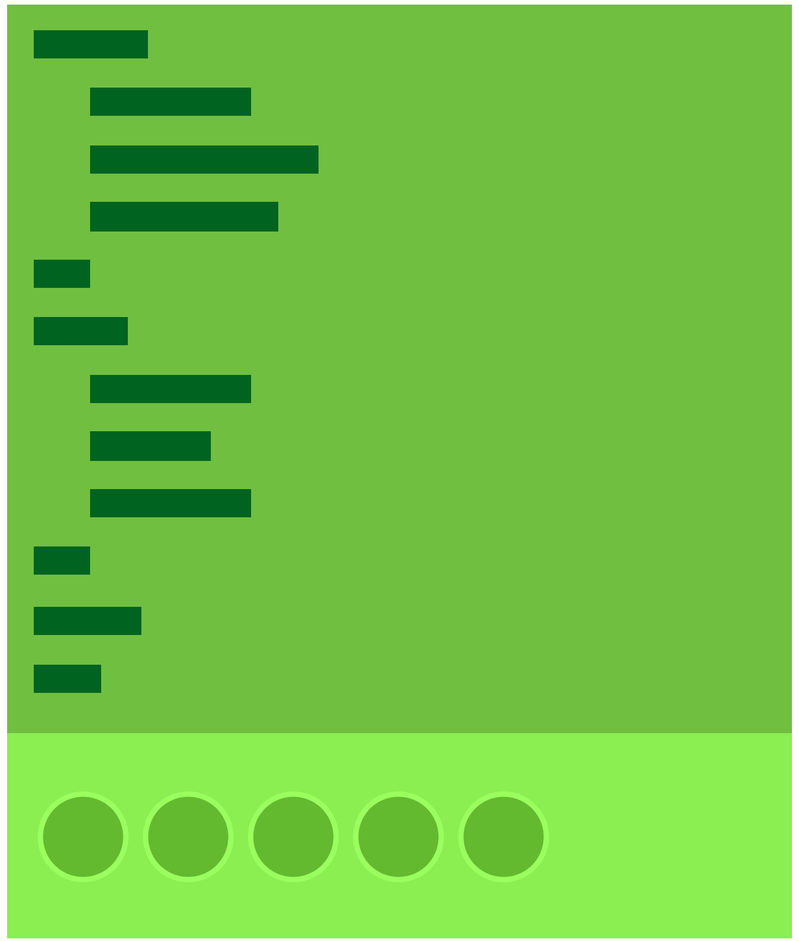
Why?

Test Quality

Coverage

Tests = Communication

Better Code



TDD Cycle

Tests, red, code, green, refactor, repeat

Why?

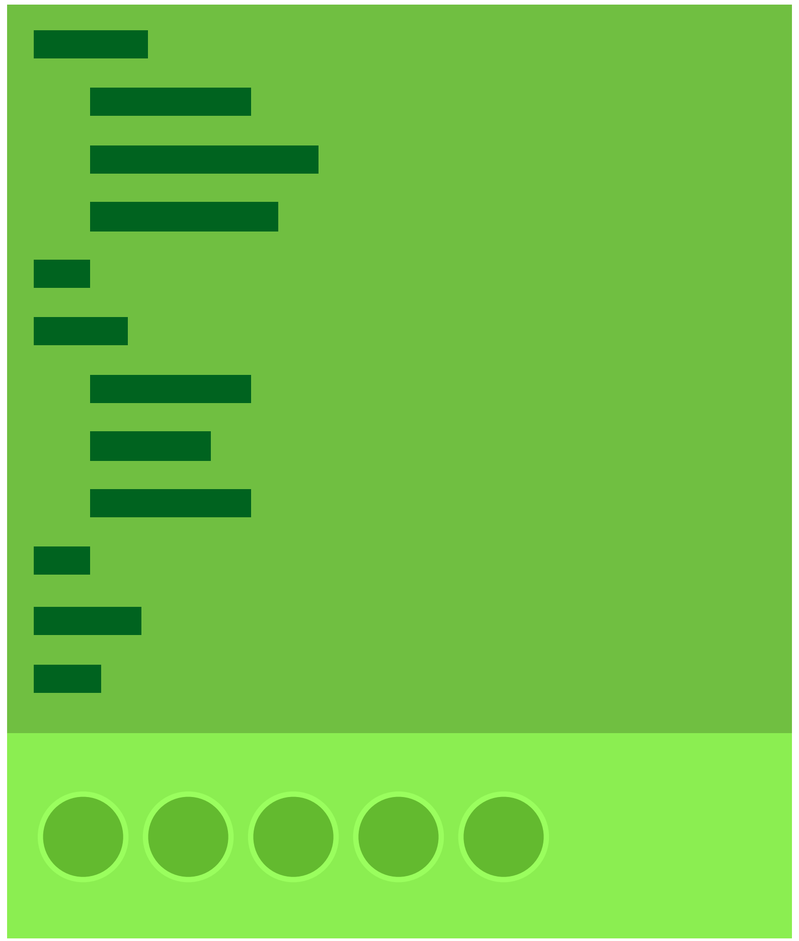
Test Quality

Coverage

Tests = Communication

Better Code

Code has to be testable



TDD Cycle

Tests, red, code, green, refactor, repeat

Why?

Test Quality

Coverage

Tests = Communication

Better Code

Code has to be testable

TDD Cycle

Tests, red, code, green, refactor, repeat

Why?

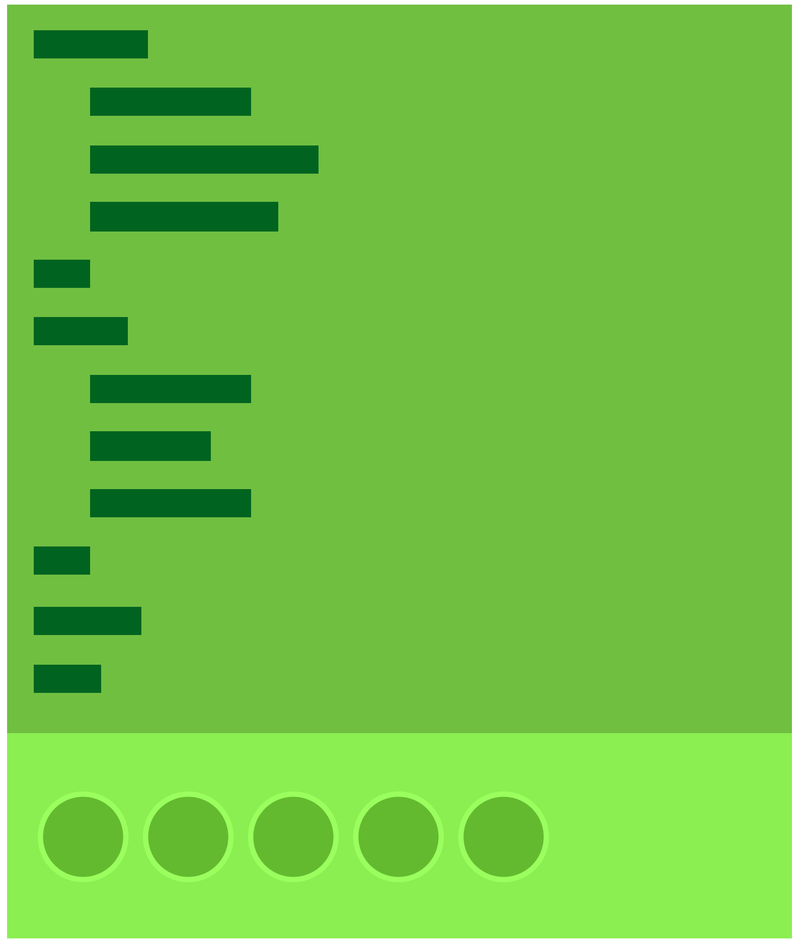
Test Quality

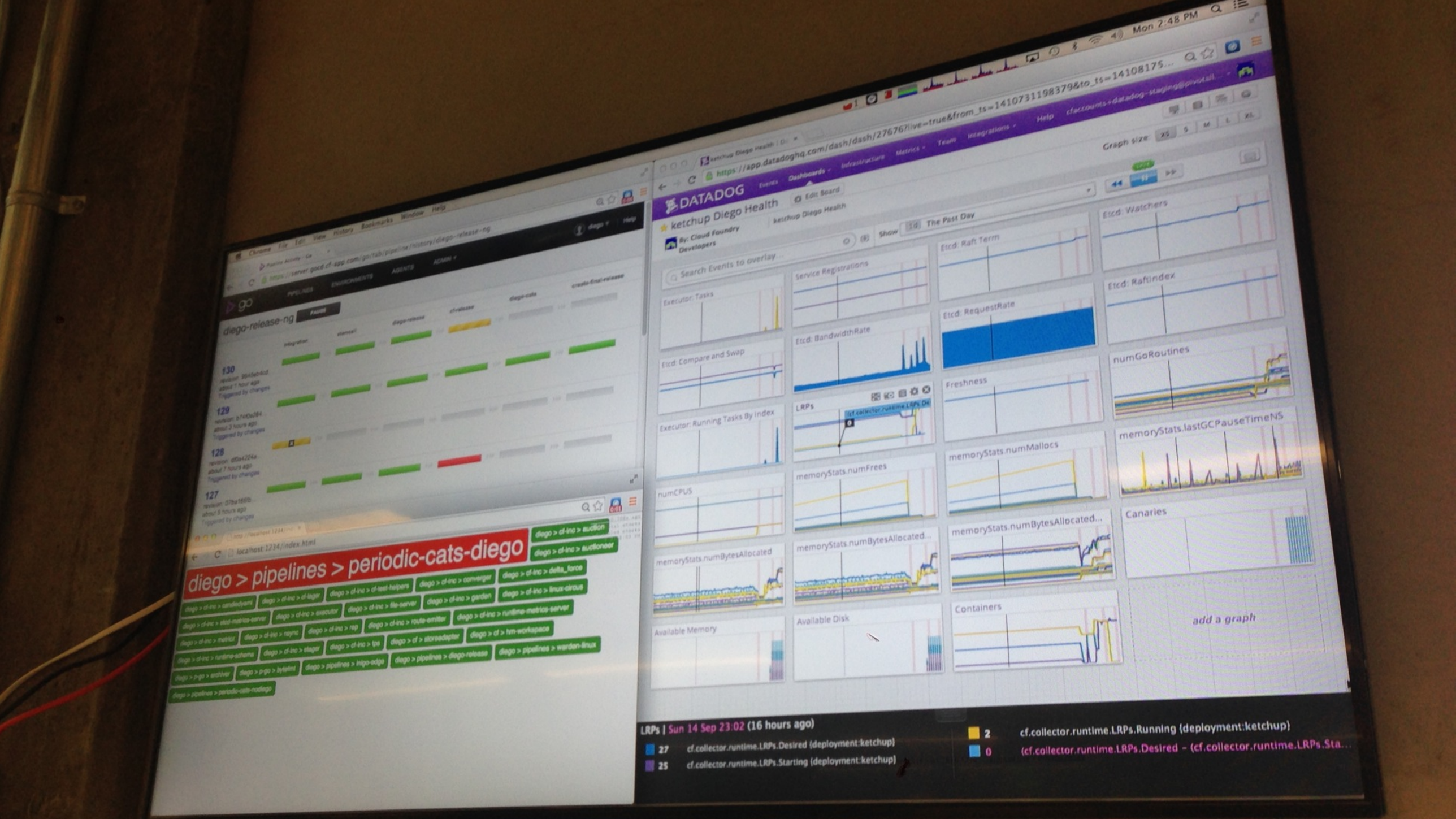
Coverage

Tests = Communication

Better Code

Code has to be testable




diego-release-ng
diego > pipelines > periodic-cats-diego
DATADOG
ketchup Diego Health
Etcd: Watchers
Etcd: Raft Term
Etcd: RaftIndex
Etcd: RequestRate
Etcd: BandwidthRate
Etcd: Compare and Swap
Service Registrations
Executor: Tasks
numGoRoutines
Freshness
LRPs
Executor: Running Tasks By Index
memoryStats.lastGCPauseTimeNS
memoryStats.numMallocs
memoryStats.numFrees
numCPUS
Canaries
memoryStats.numBytesAllocated...
memoryStats.numBytesAllocated
Containers
Available Disk
Available Memory
add a graph
LRPs | Sun 14 Sep 23:02 (16 hours ago)
27 cf.collector.runtime.LRPs.Desired {deployment:ketchup}
25 cf.collector.runtime.LRPs.Starting {deployment:ketchup}
2 cf.collector.runtime.LRPs.Running {deployment:ketchup}
0 (cf.collector.runtime.LRPs.Desired - (cf.collector.runtime.LRPs.Sta...


EXIT

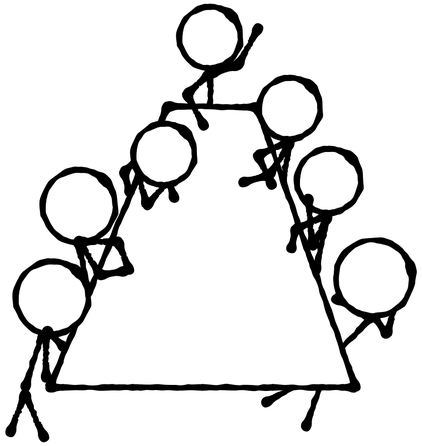
Pivotal™

culture

Pairing

Testing

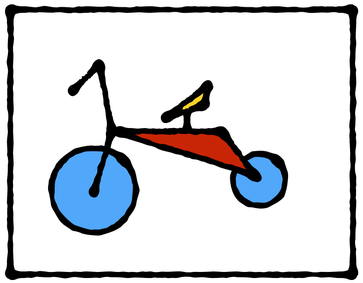
Agile Planning



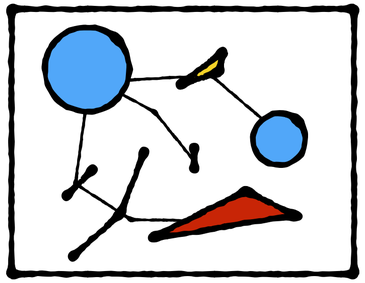
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

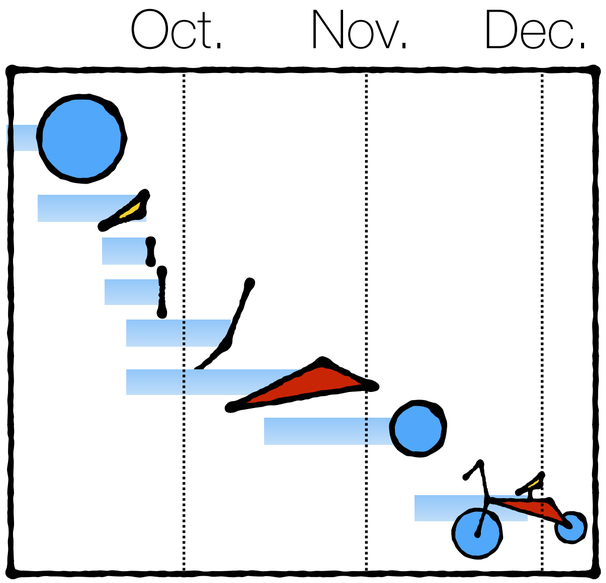
Define
Requirements



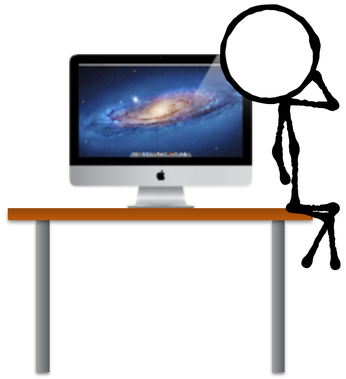
Architect/
Design



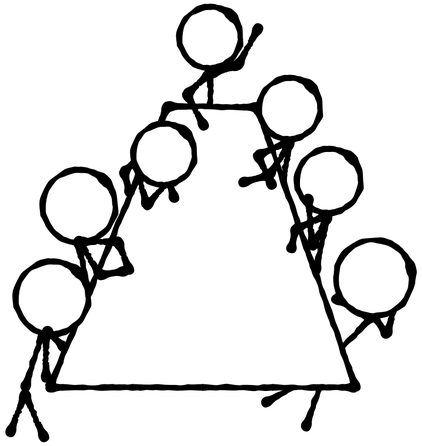
Determine
Dependencies



Estimate Effort
Budget Time



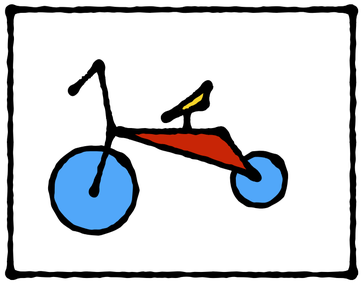
Develop, develop, develop…



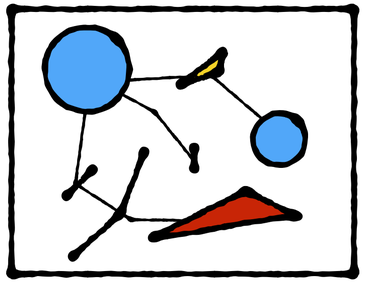
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

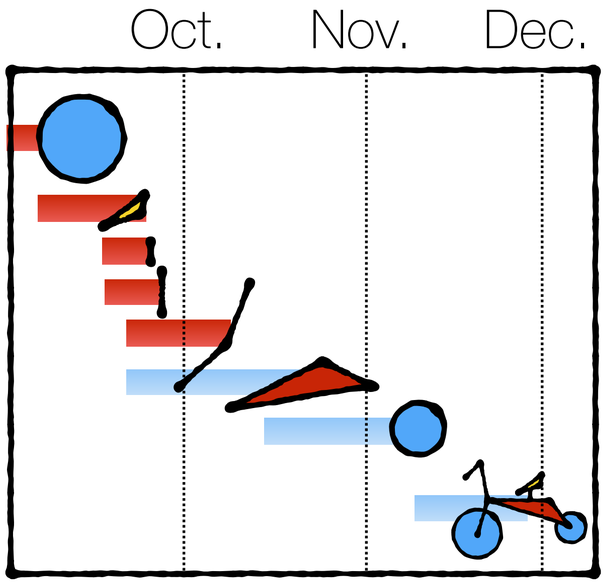
Define
Requirements



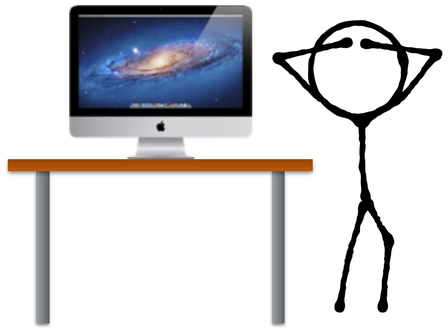
Architect/
Design



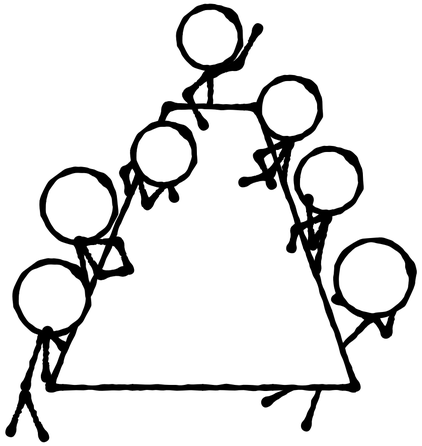
Determine
Dependencies



Estimate Effort
Budget Time



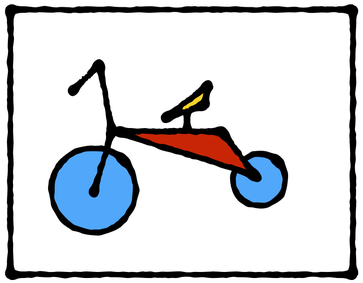
Develop, develop, develop…



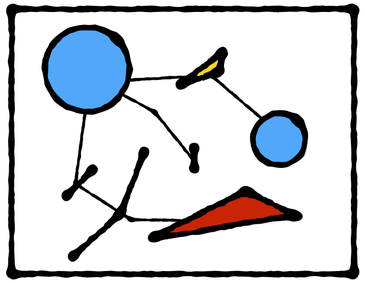
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

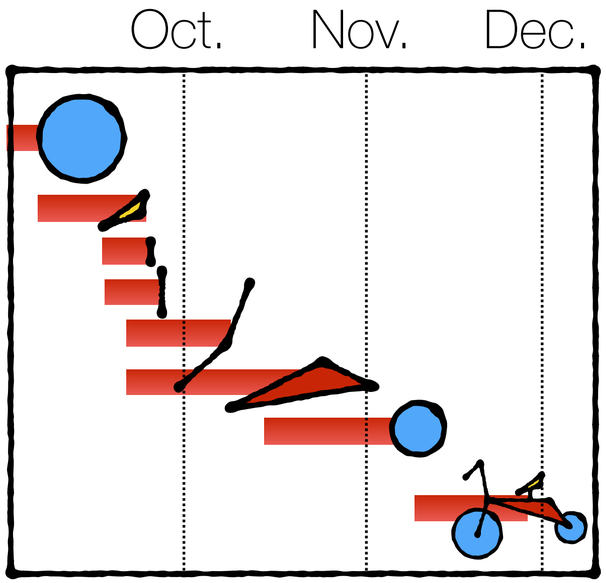
Define
Requirements



Architect/
Design

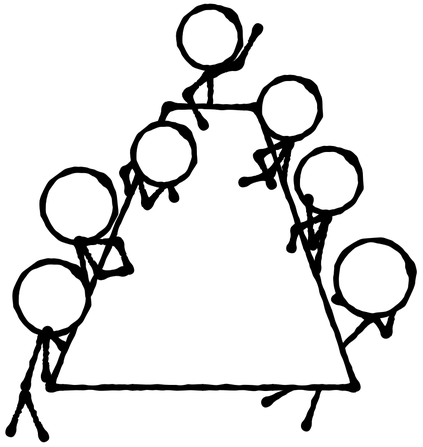


Determine
Dependencies



Estimate Effort
Budget Time

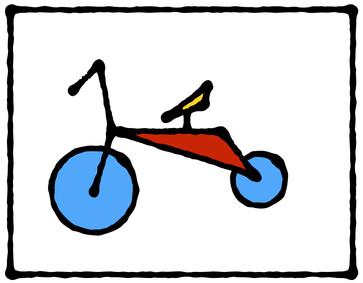
Develop, develop, develop…



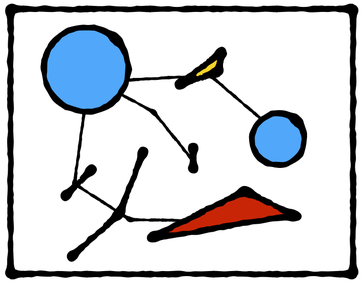
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

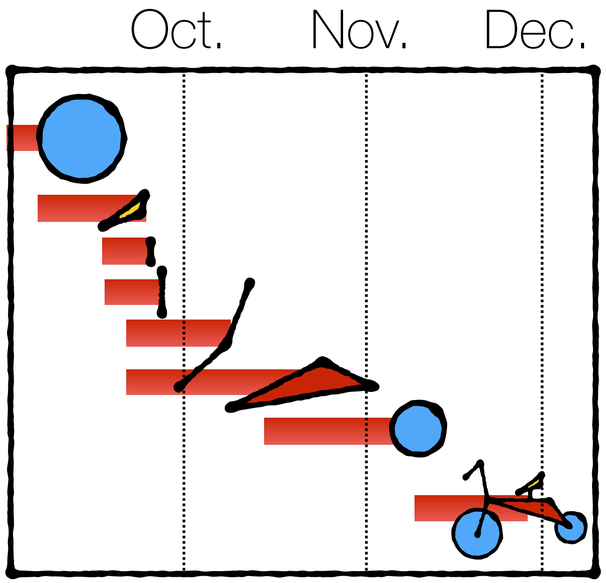
Define
Requirements



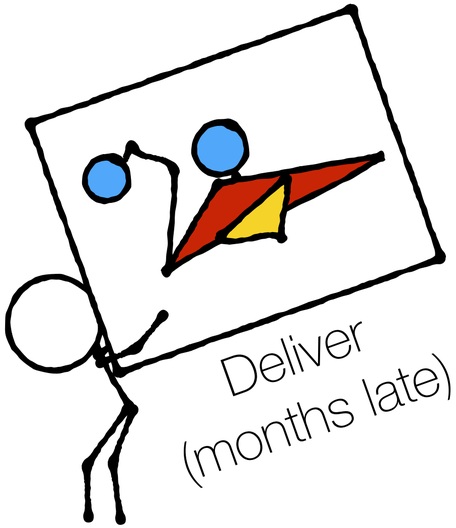
Architect/
Design



Determine
Dependencies

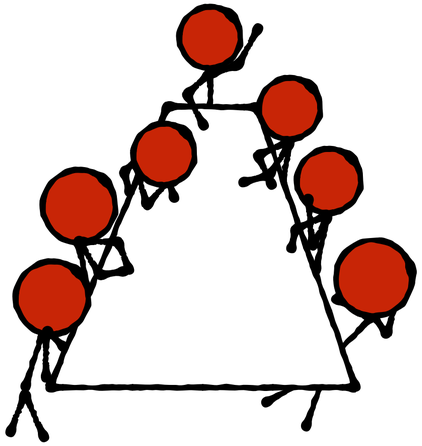


Estimate Effort
Budget Time



Deliver
(months late)

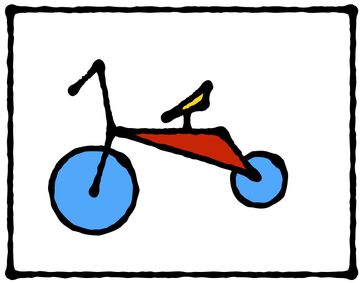
Develop, develop, develop…



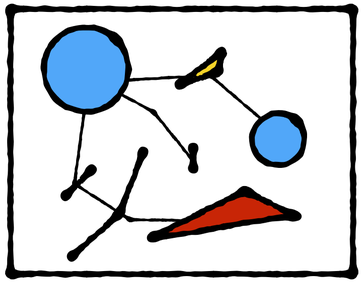
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

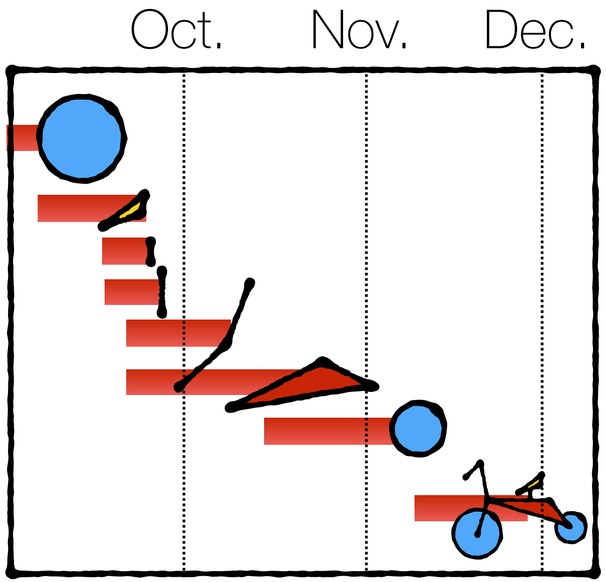
Define
Requirements



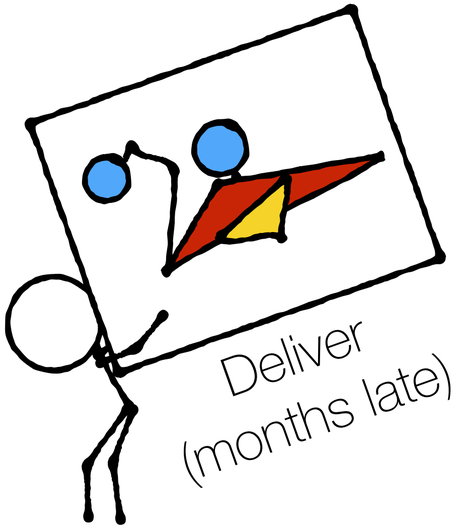
Architect/
Design



Determine
Dependencies



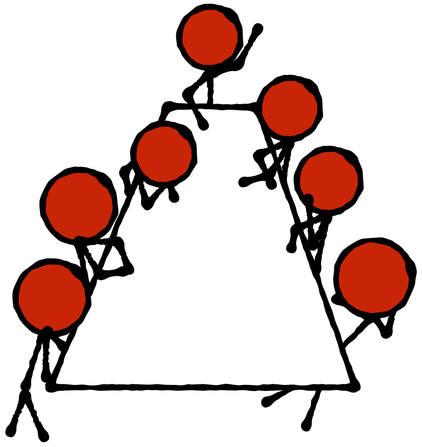
Estimate Effort
Budget Time



Deliver
(months late)

Develop, develop, develop…

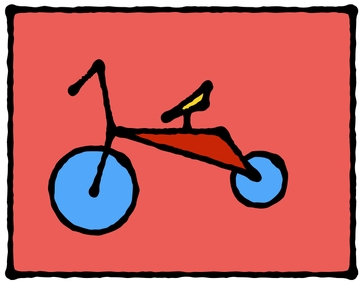
Upfront planning divorced from reality



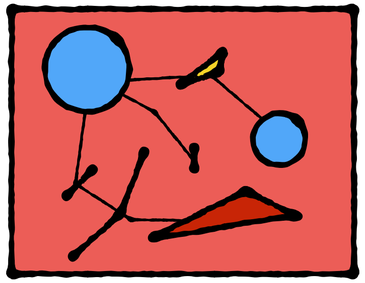
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

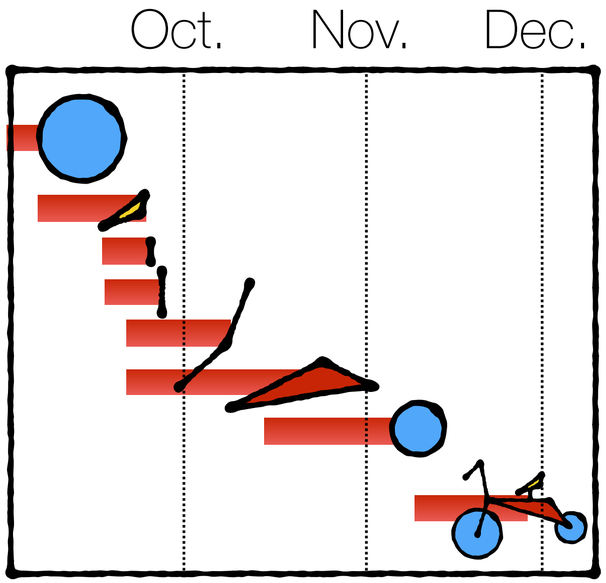
Define
Requirements



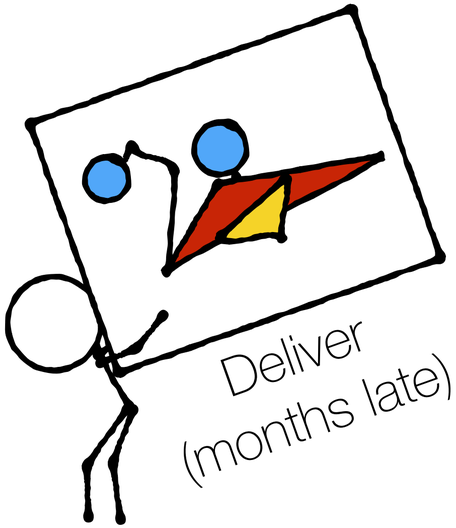
Architect/
Design



Determine
Dependencies



Estimate Effort
Budget Time

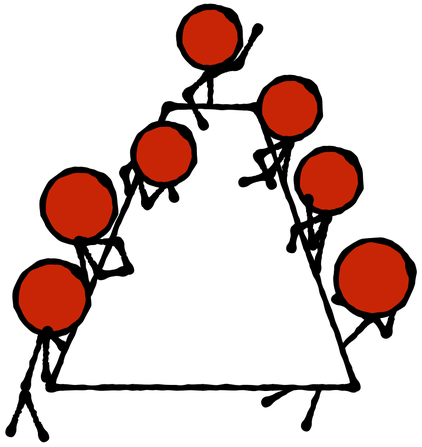


Deliver
(months late)

Develop, develop, develop…

Upfront planning divorced from reality

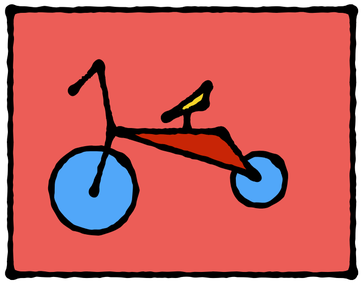
Artificial deadlines divorced from reality



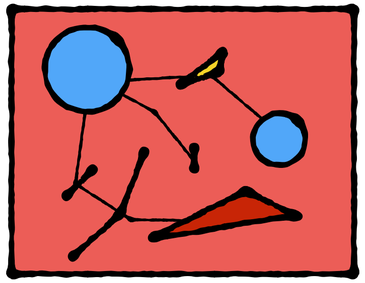
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

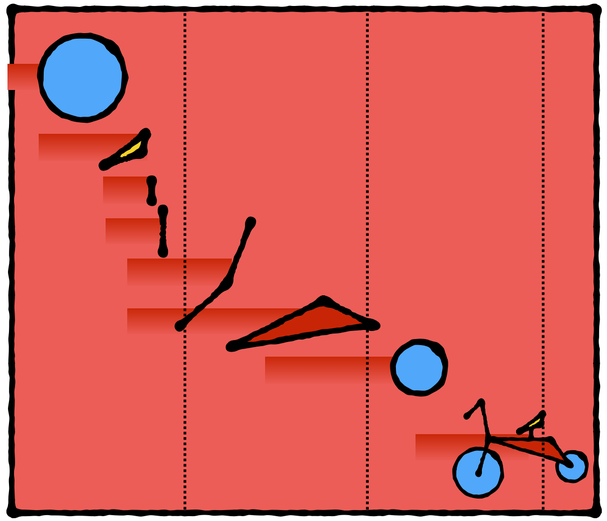
Define
Requirements



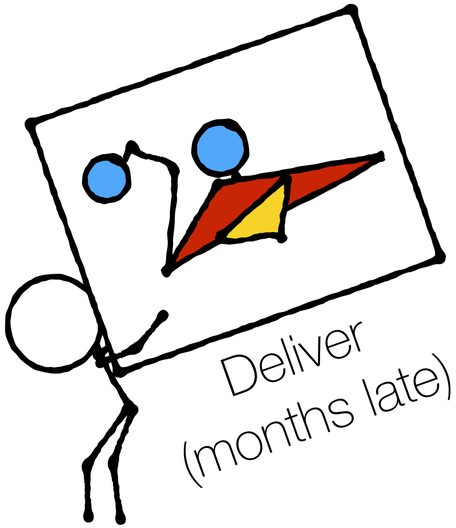
Architect/
Design



Determine
Dependencies



Estimate Effort
Budget Time



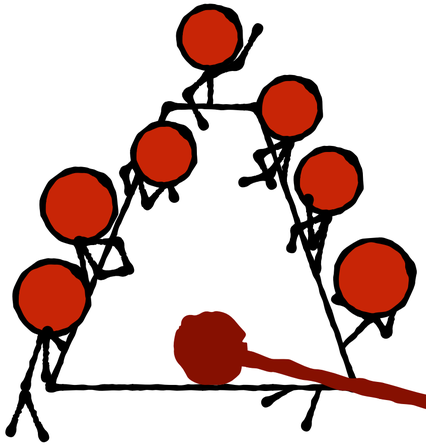
Deliver
(months late)

Develop, develop, develop…

Upfront planning divorced from reality

Artificial deadlines divorced from reality

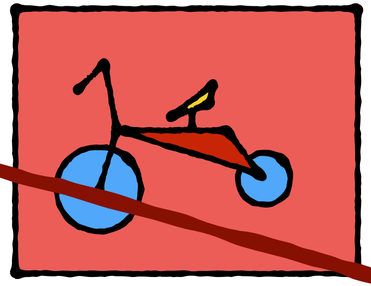
Long feedback cycle



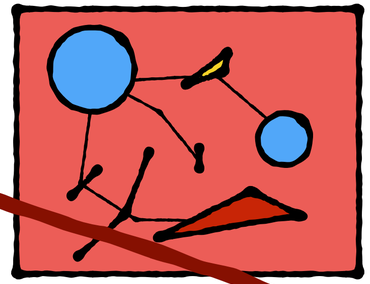
Stakeholders

A→B
Human-Powered
Edgy but chic
Target market:
recovering gen-X

Define
Requirements

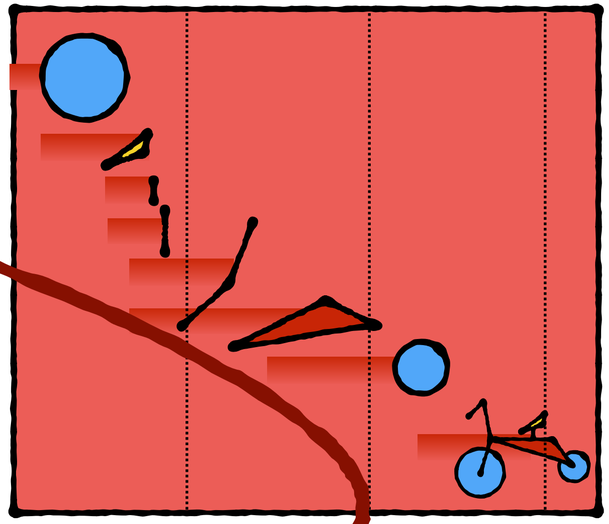


Architect/
Design

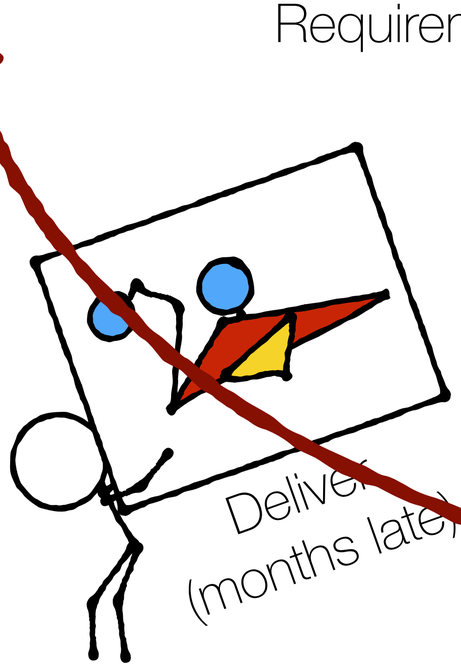


Determine
Dependencies

Oct. Nov. Dec.



Estimate Effort
Budget Time

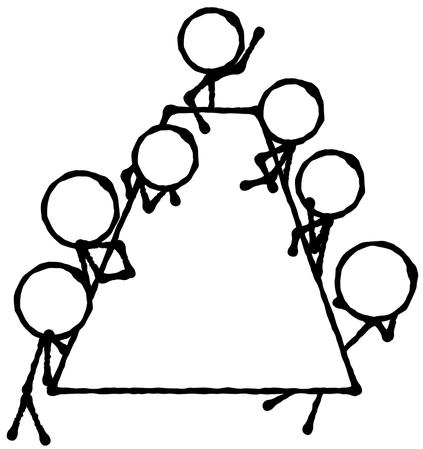


Deliver
(months late)

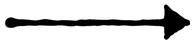
Develop, develop, develop…

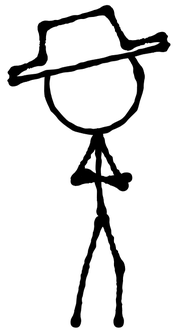
PivotalTracker

Agile, Data-Driven, Product Management

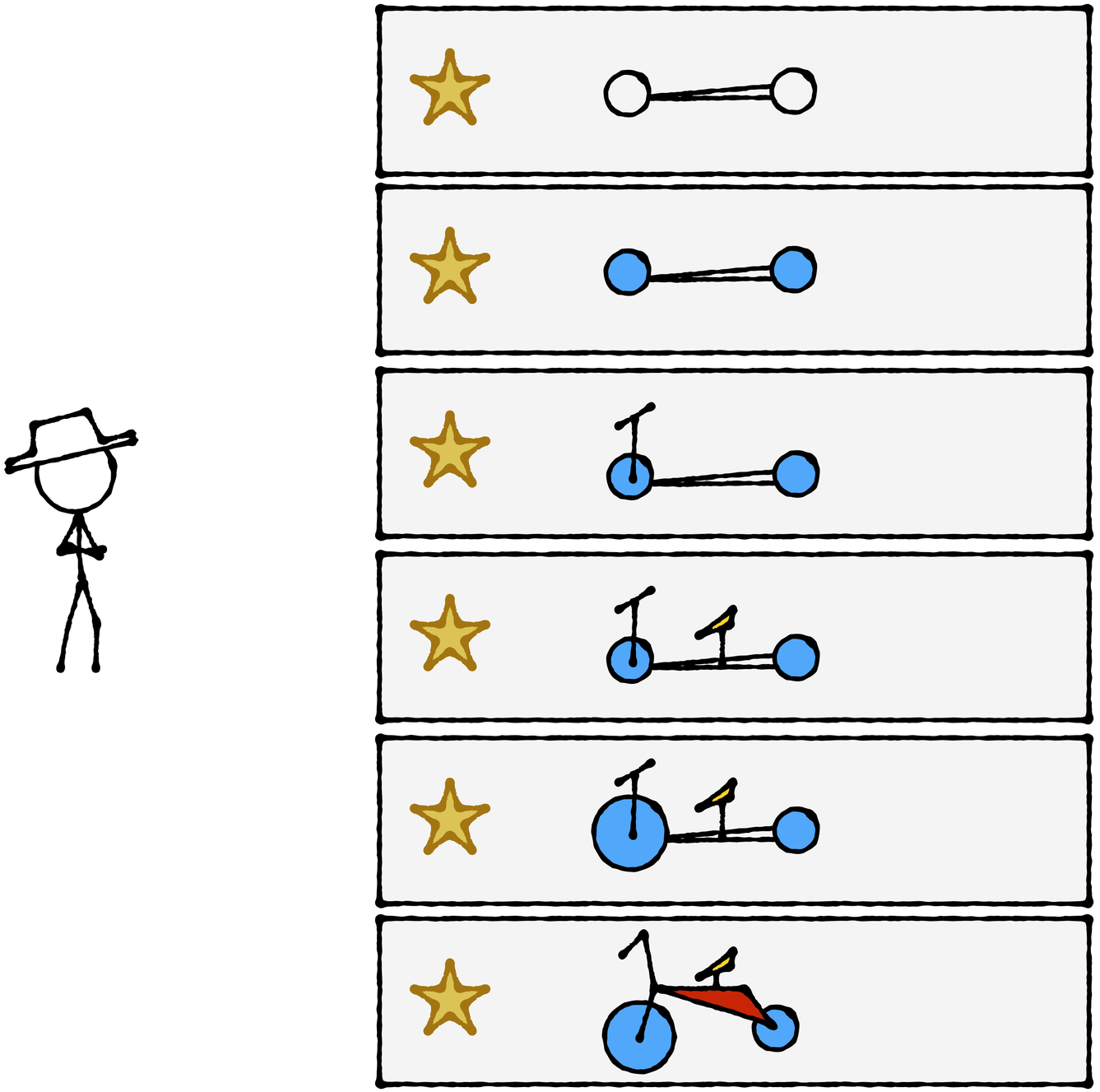


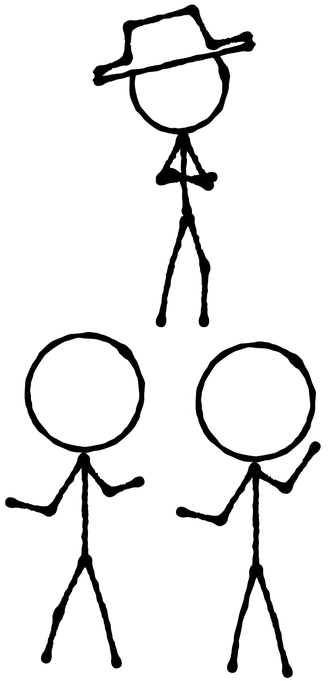
Stakeholders

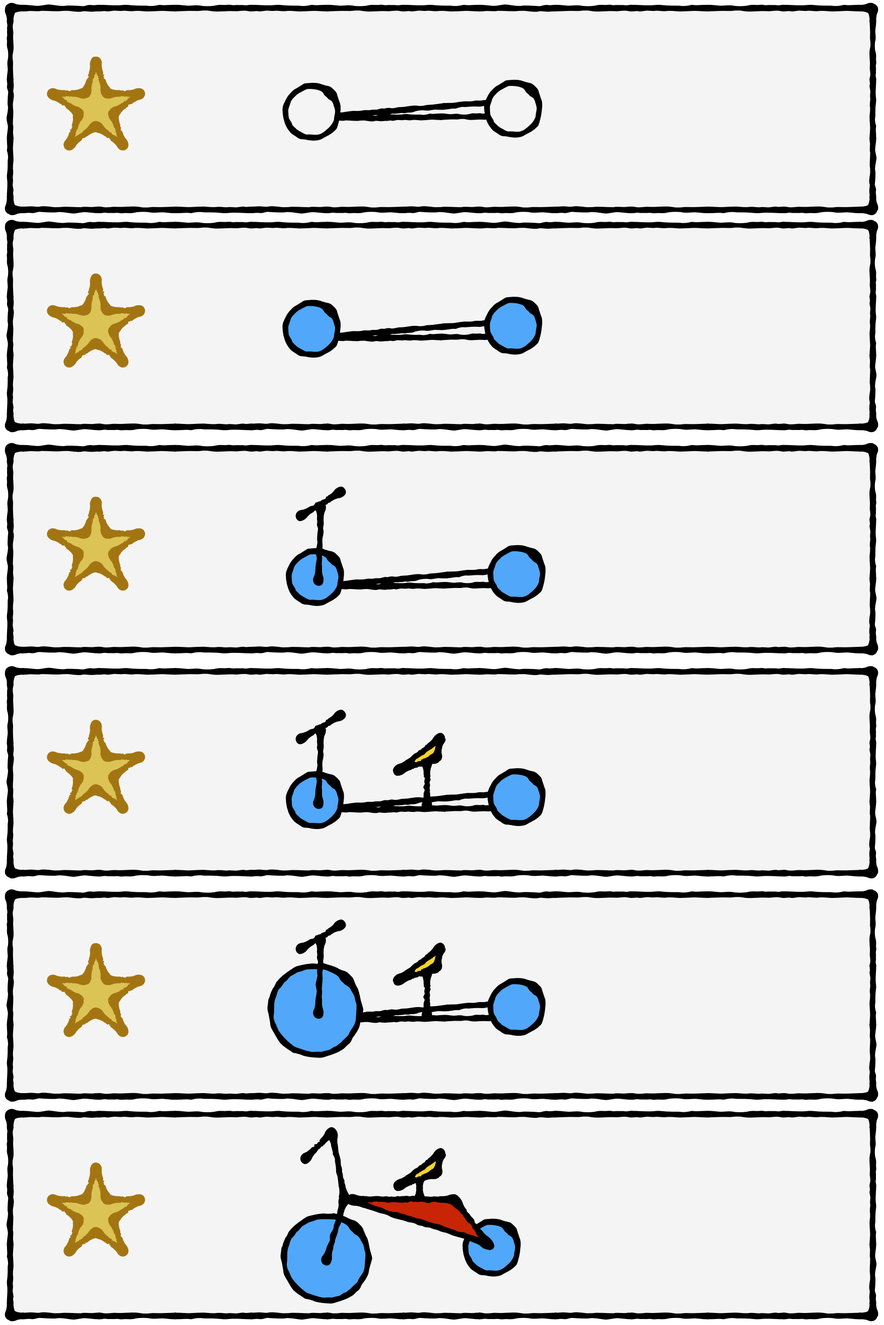


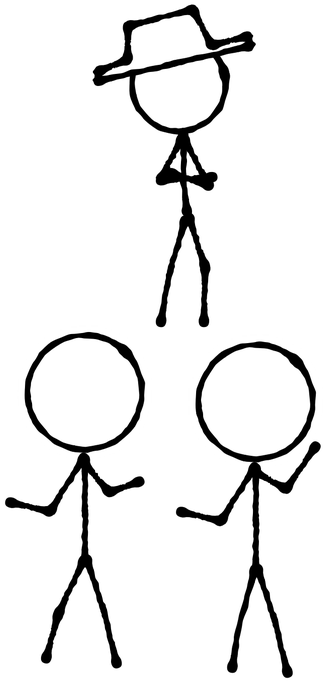


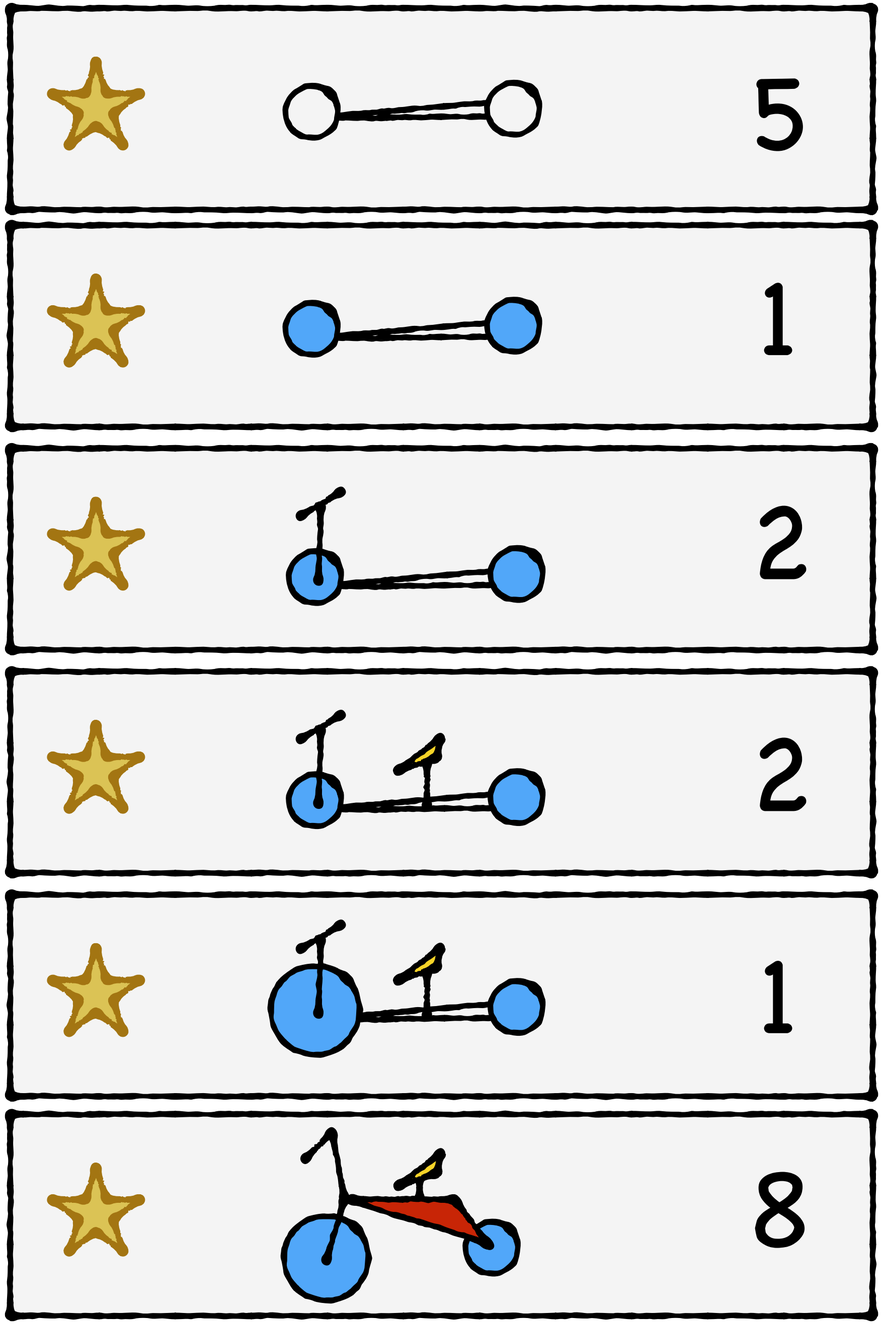
Product
Manager

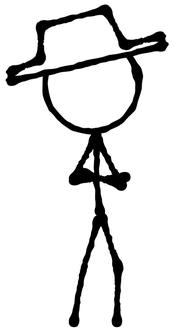


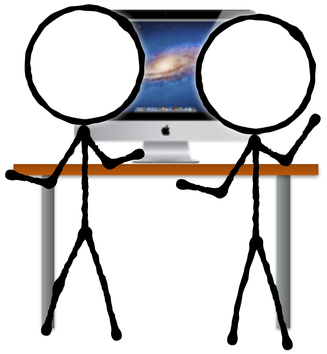


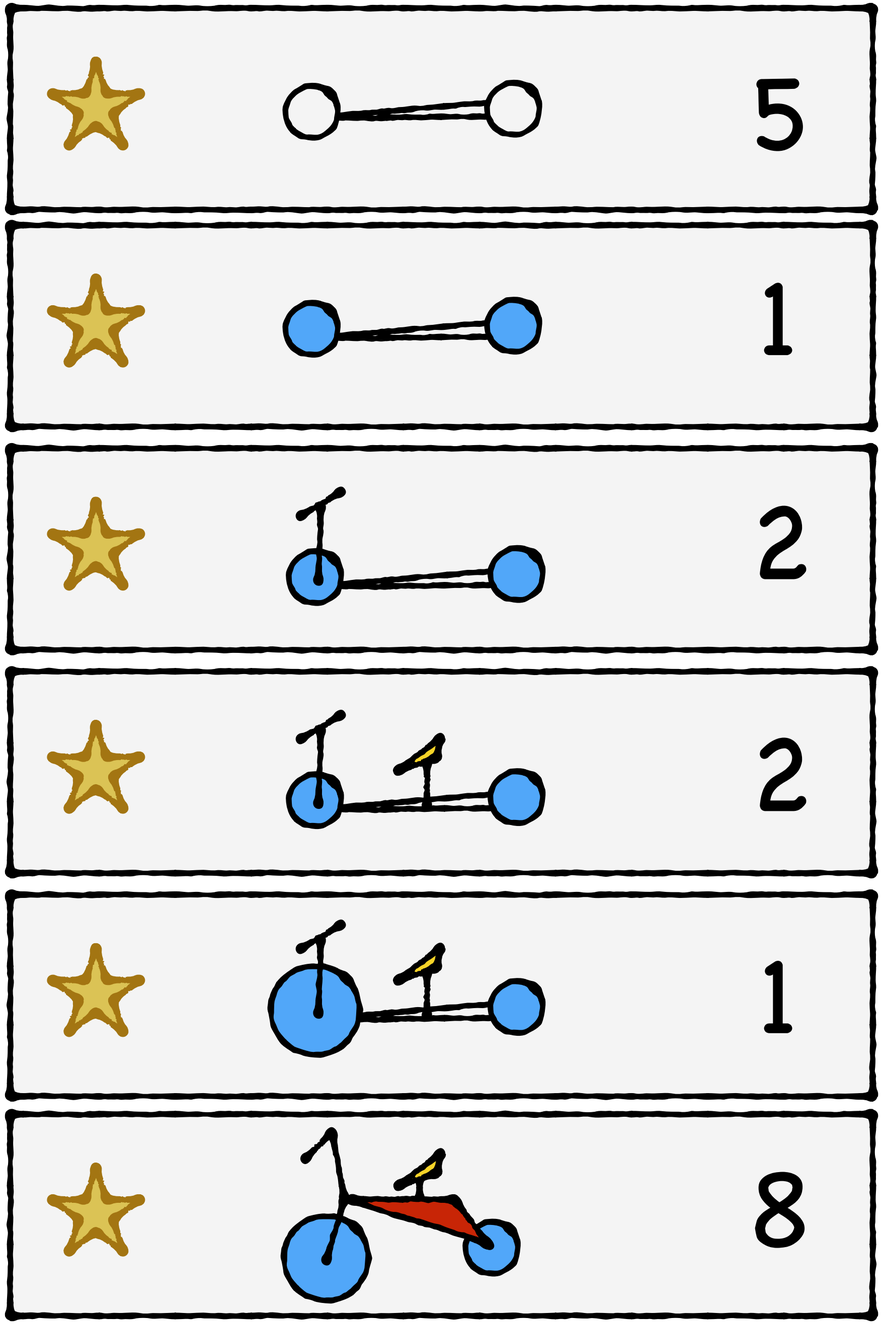


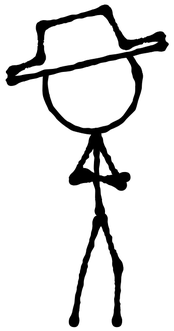


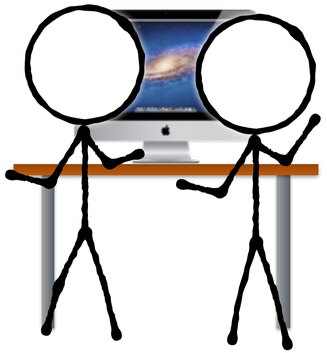

5
1
2
2
1
8

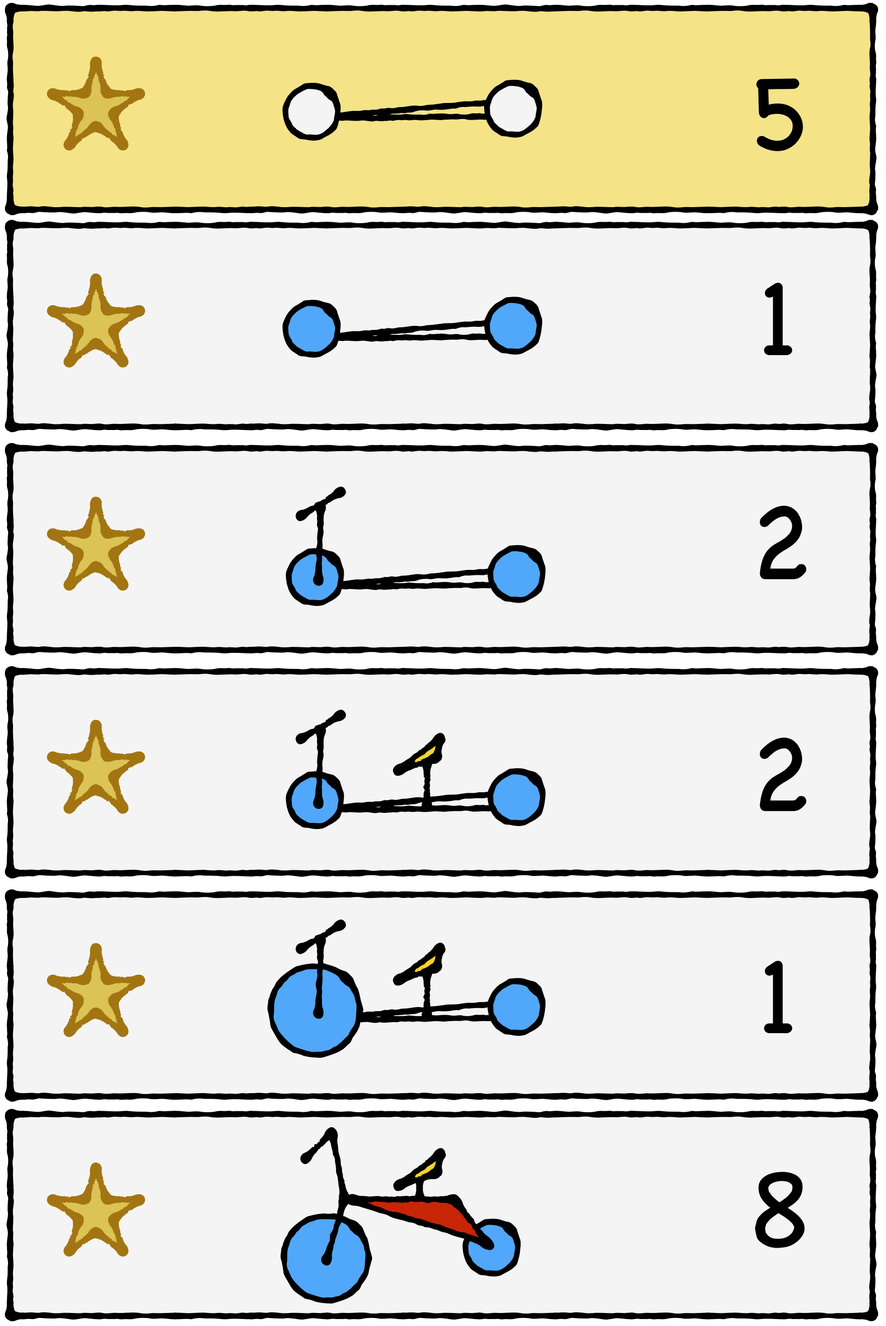


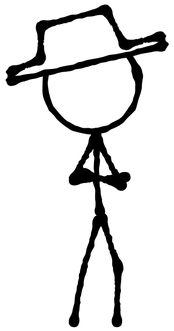


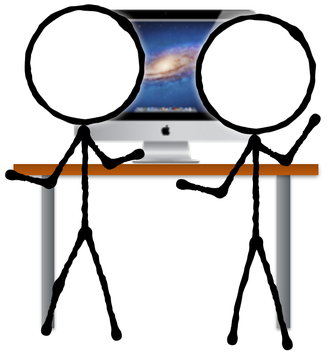

5
1
2
2
1
8

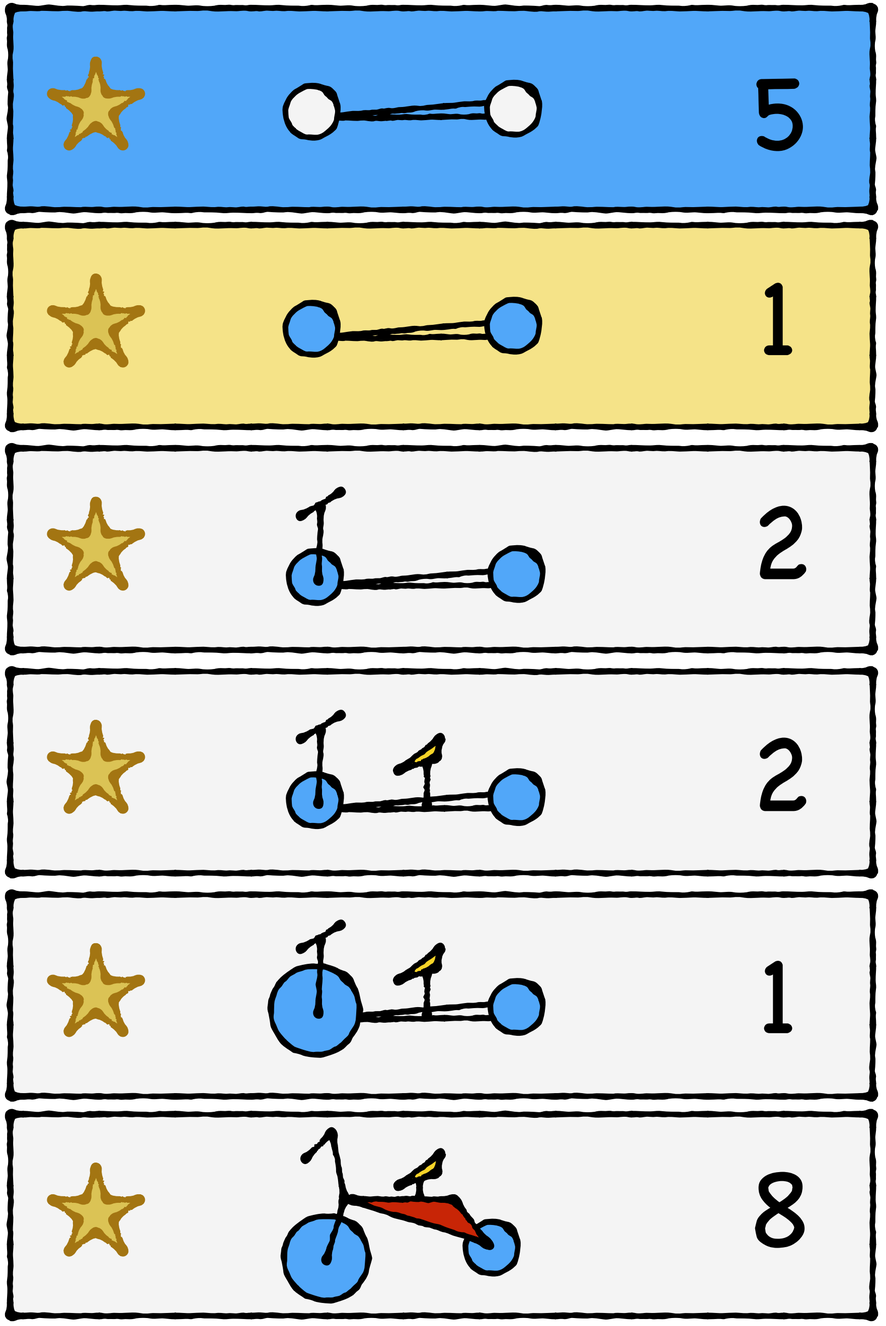


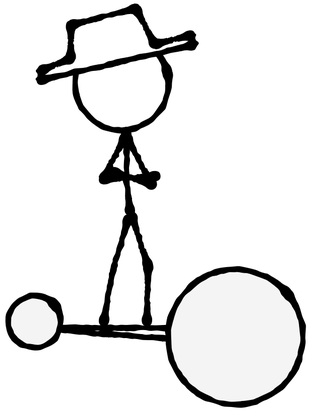


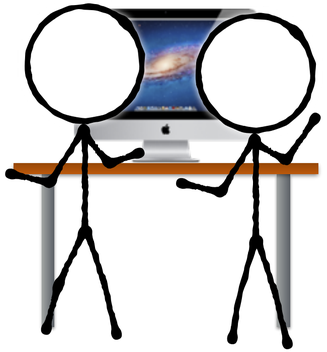

5
1
2
2
1
8

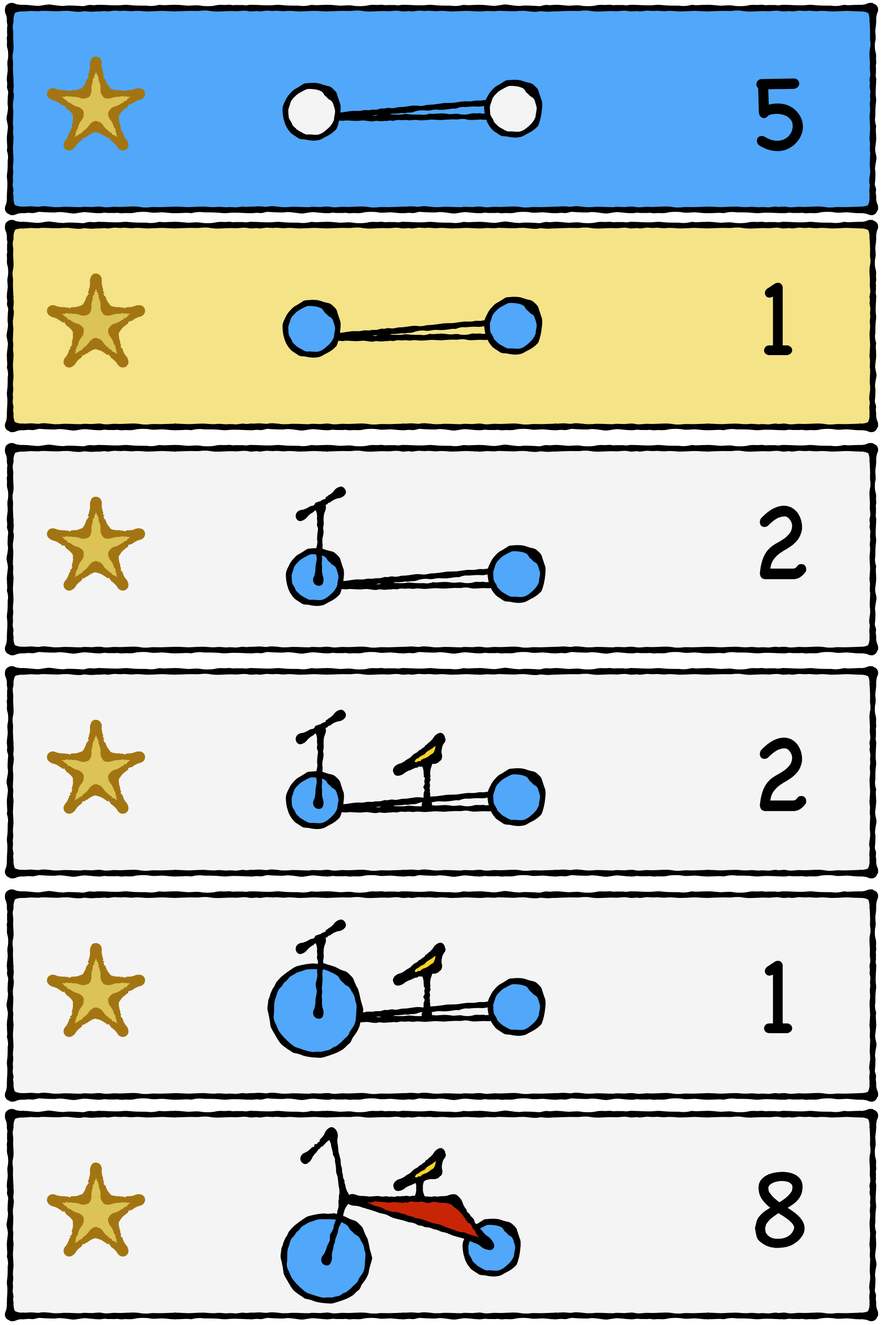


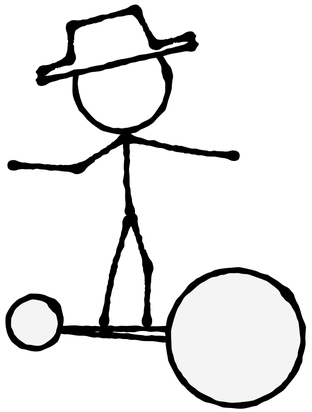


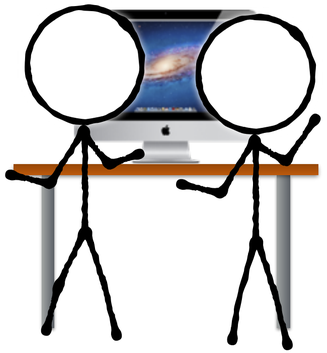

5
1
2
2
1
8

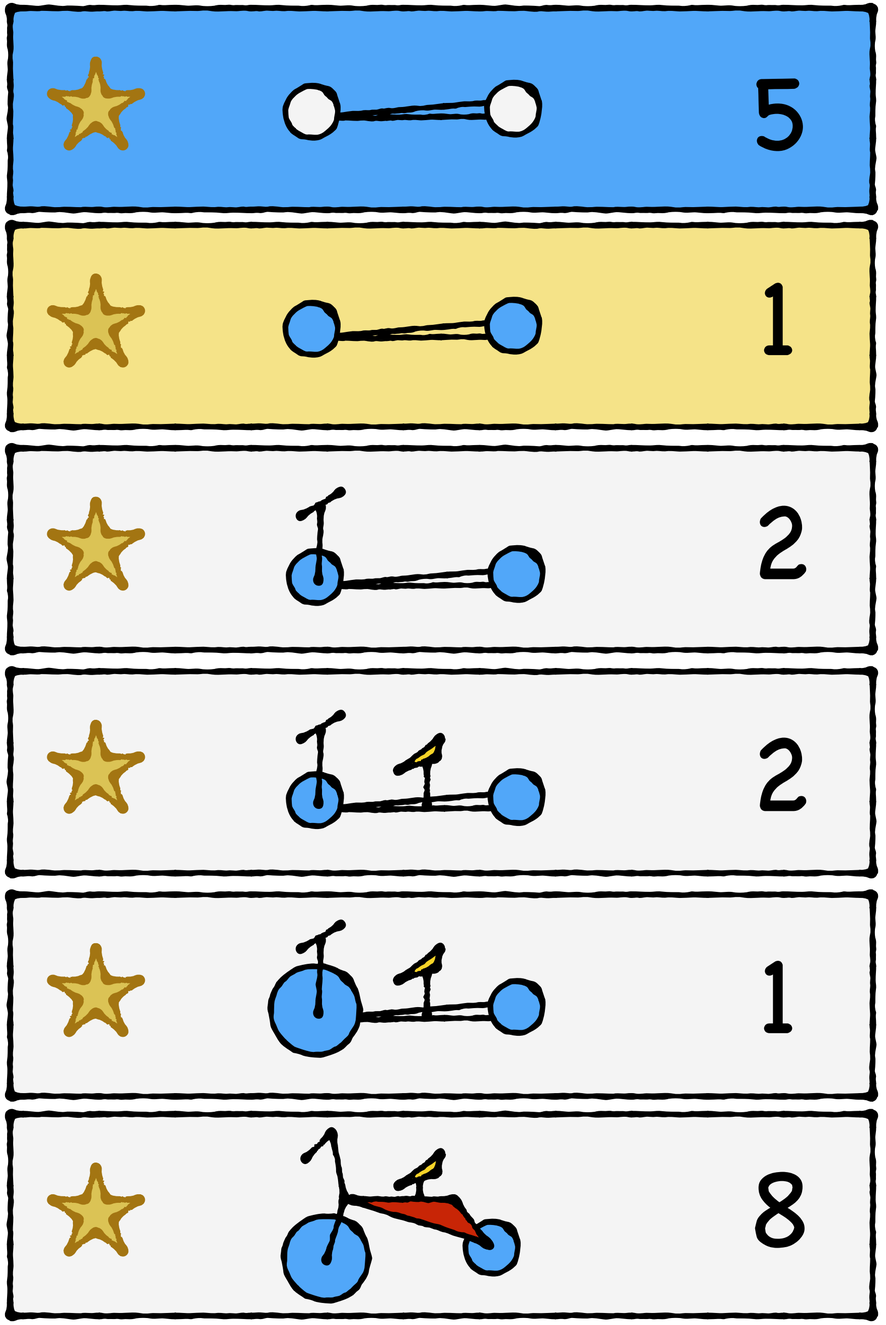


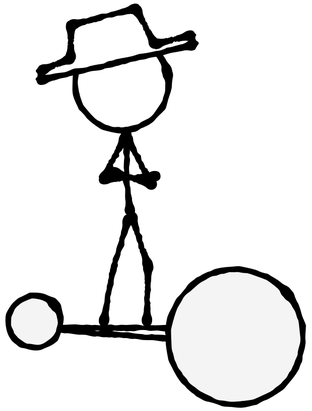


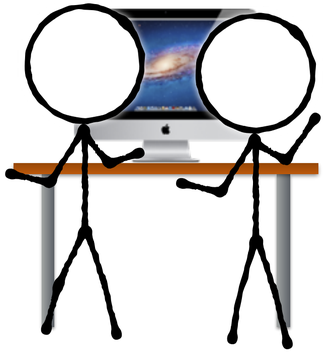

5
1
2
2
1
8

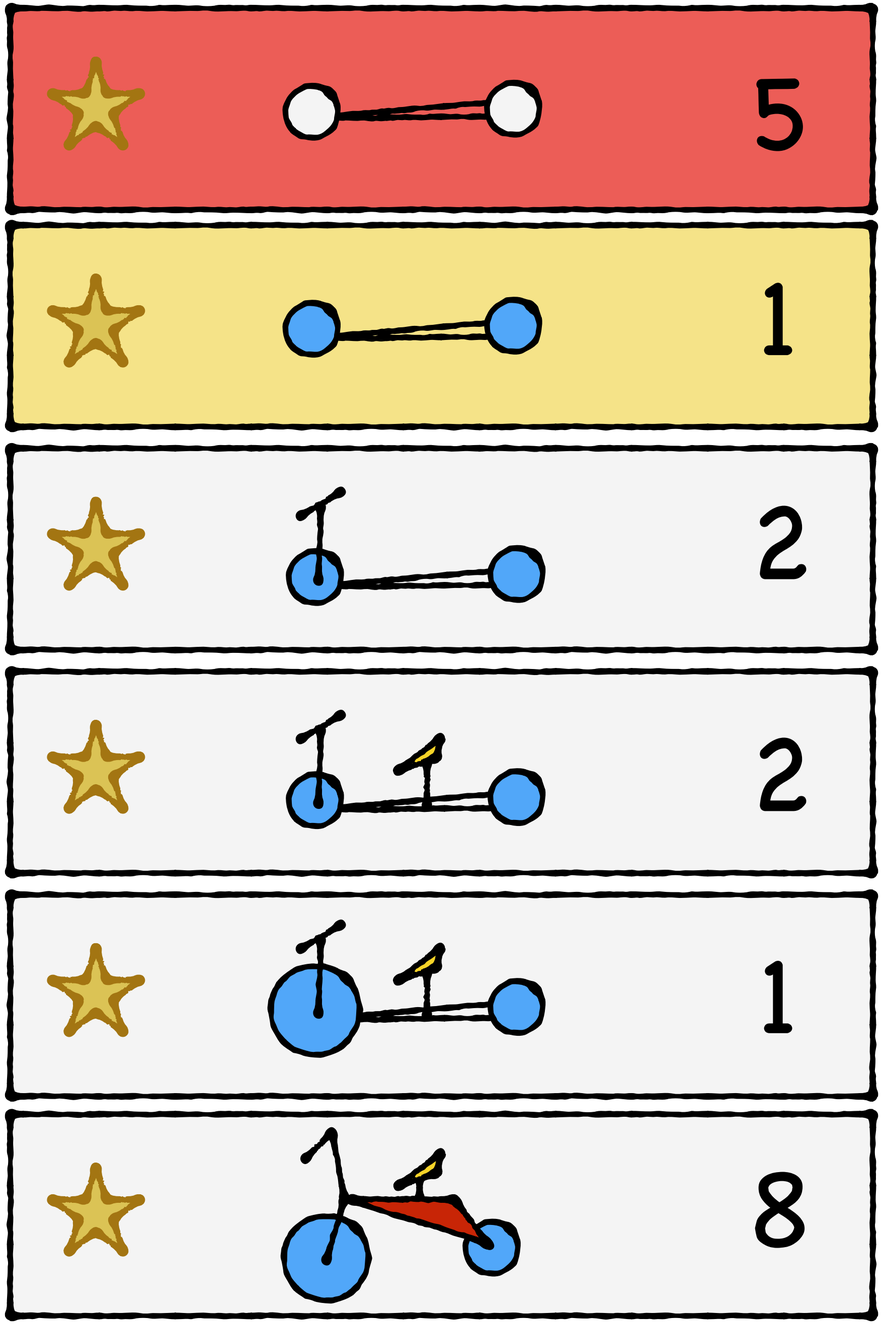


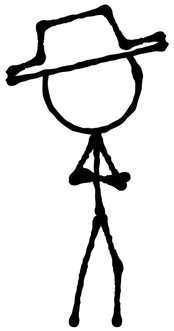


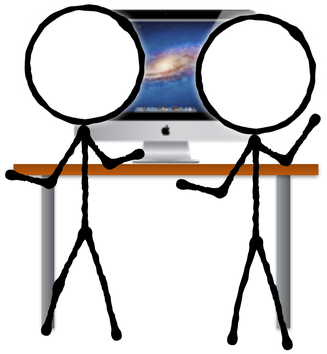

5
1
2
2
1
8

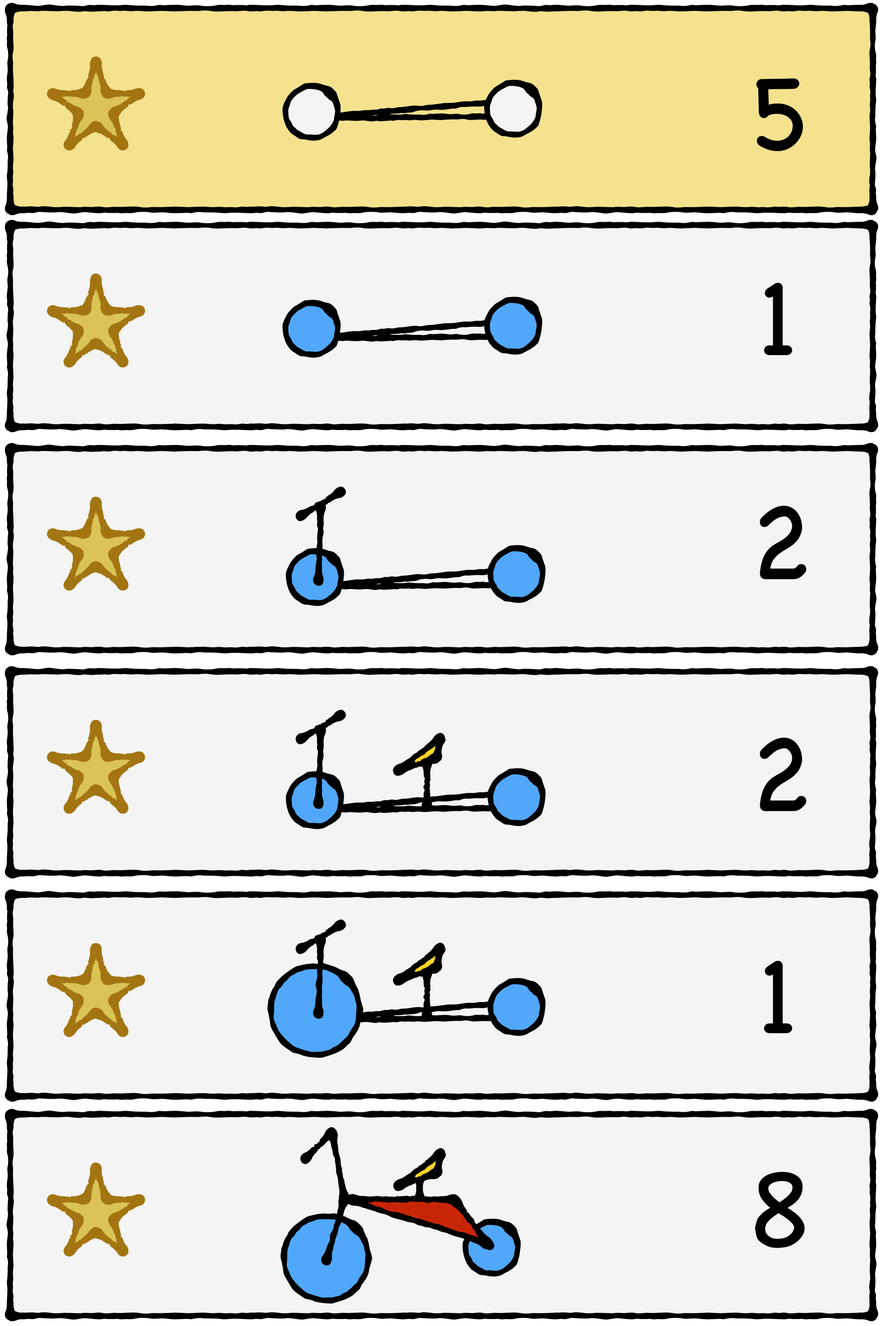


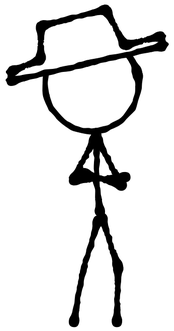


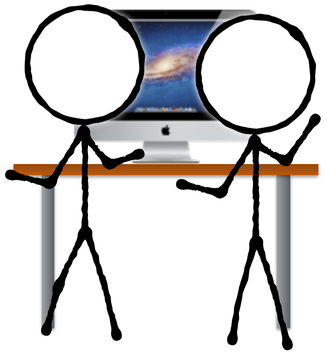

5
1
2
2
1
8

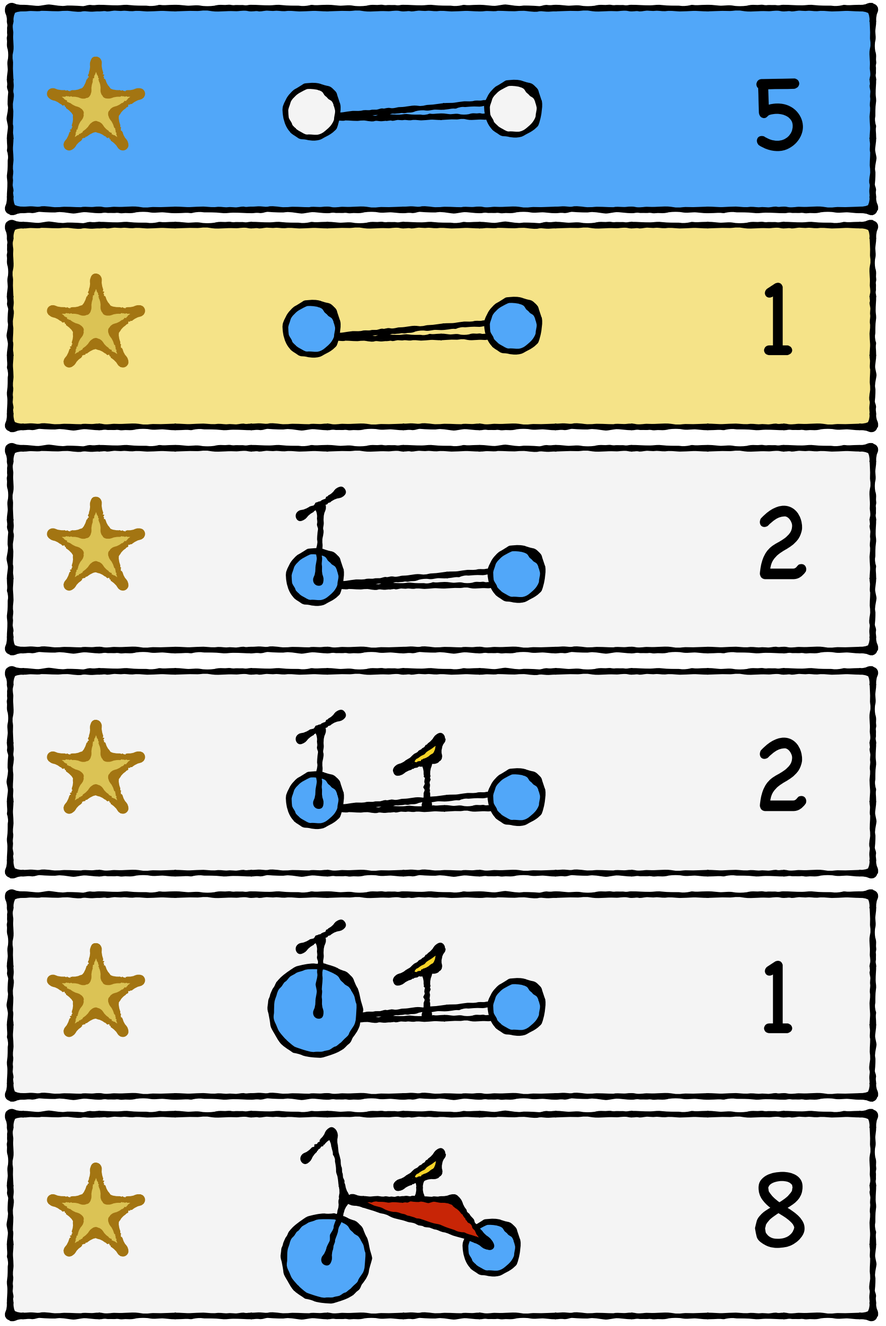


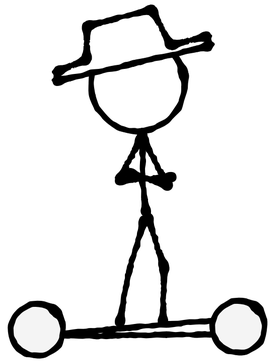


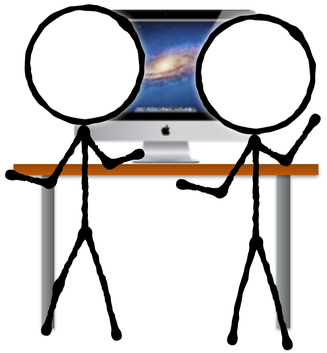

5
1
2
2
1
8

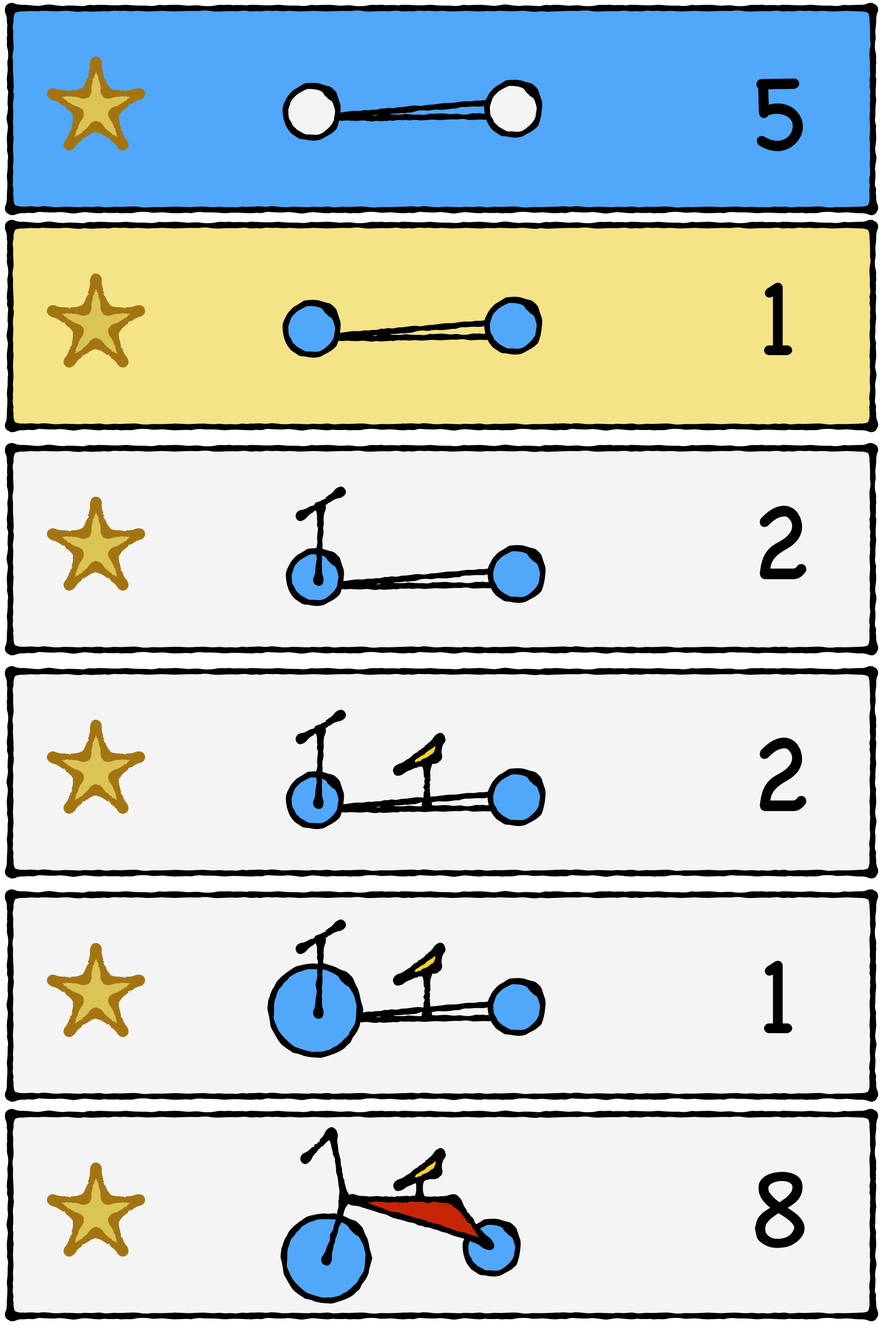


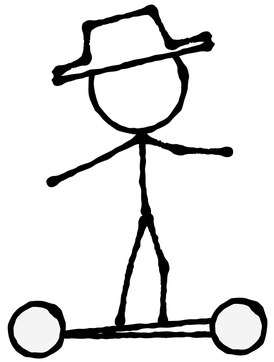


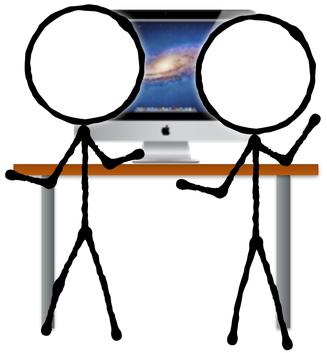

5
1
2
2
1
8

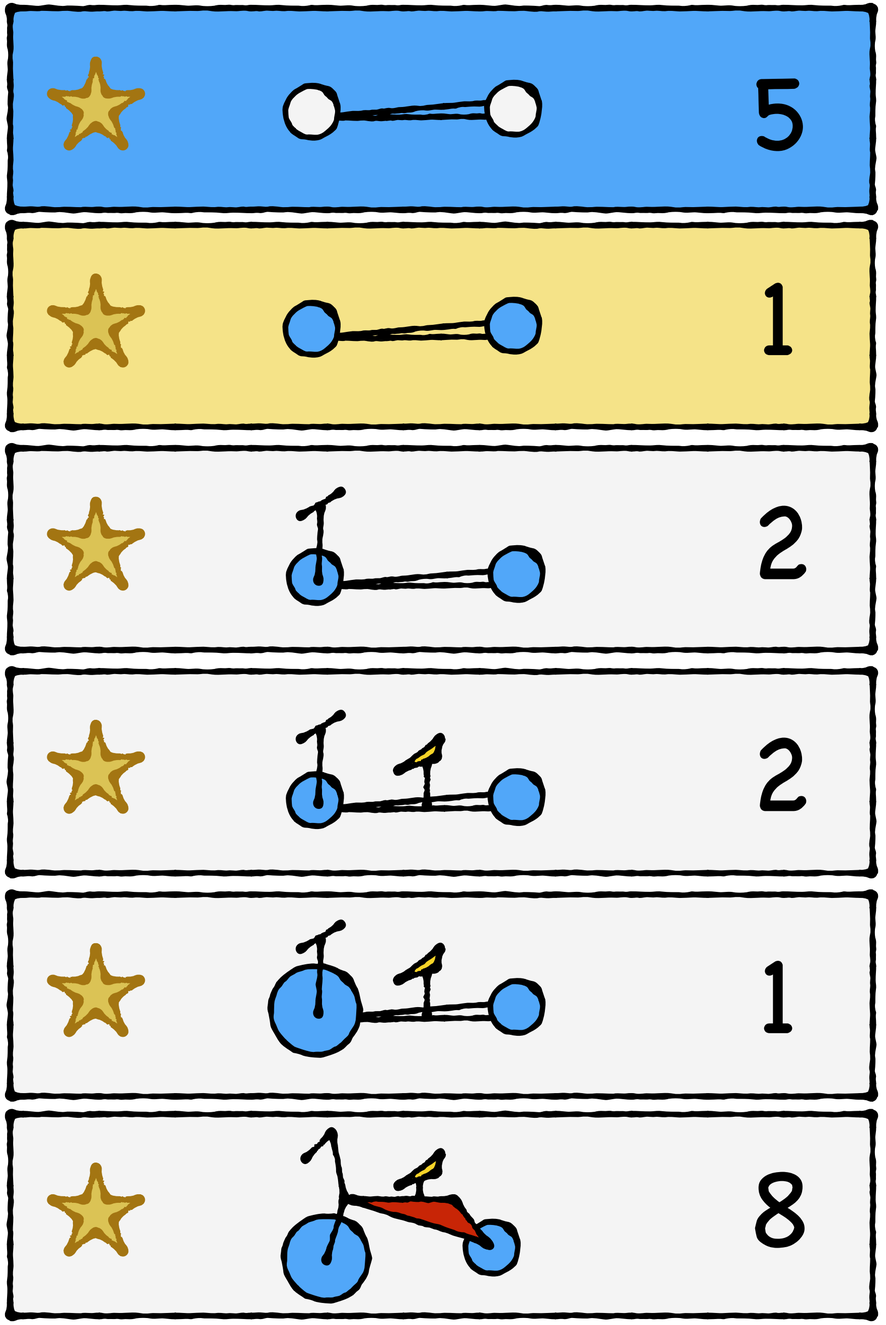




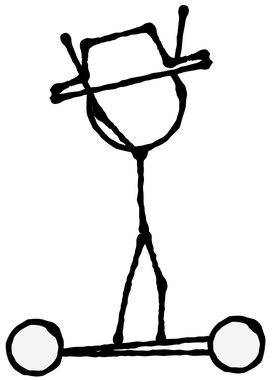

5
1
2
2
1
8

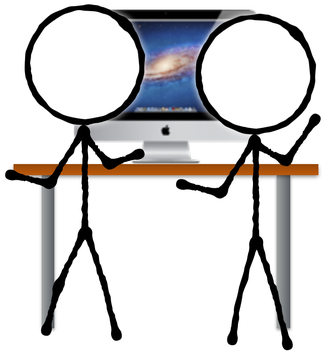


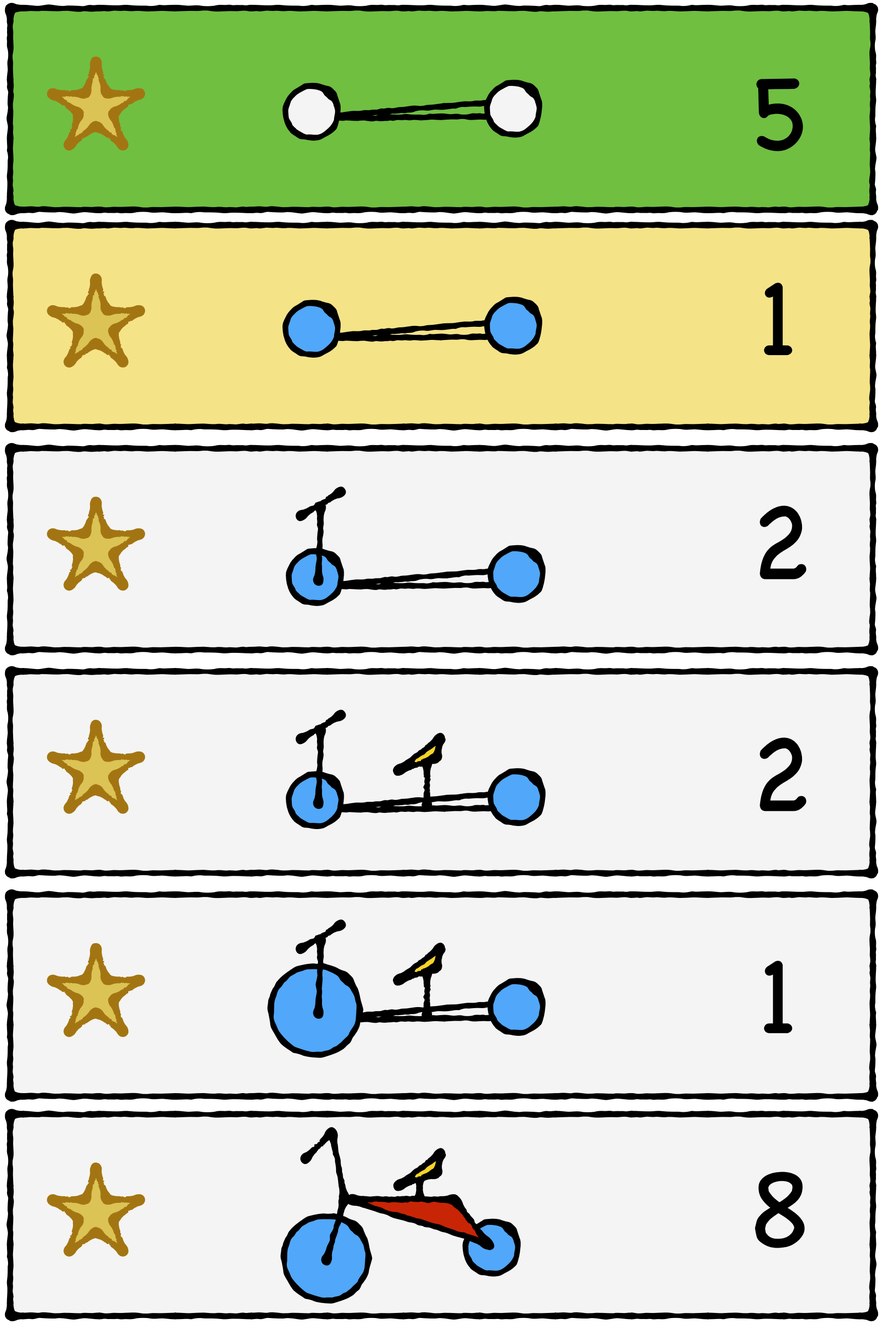



5
1
2
2
1
8

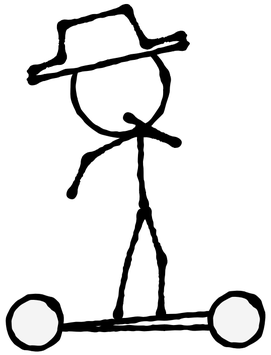
Short feedback cycles

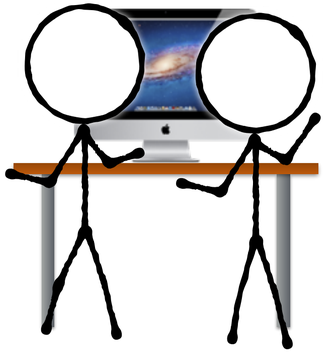


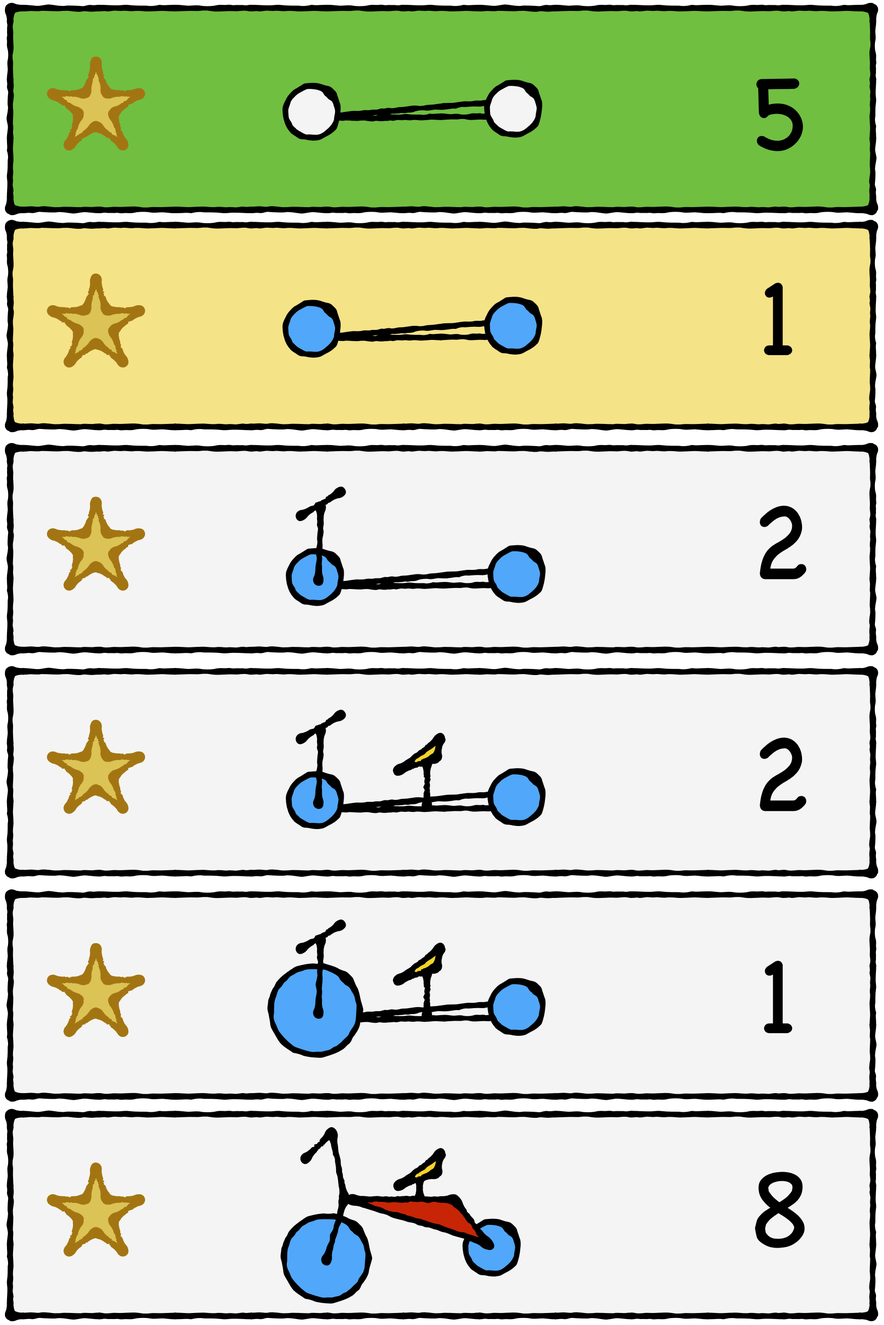




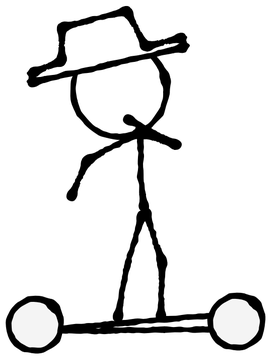
Short feedback cycles

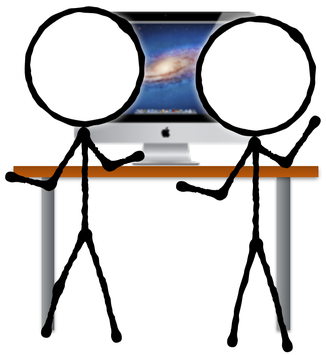


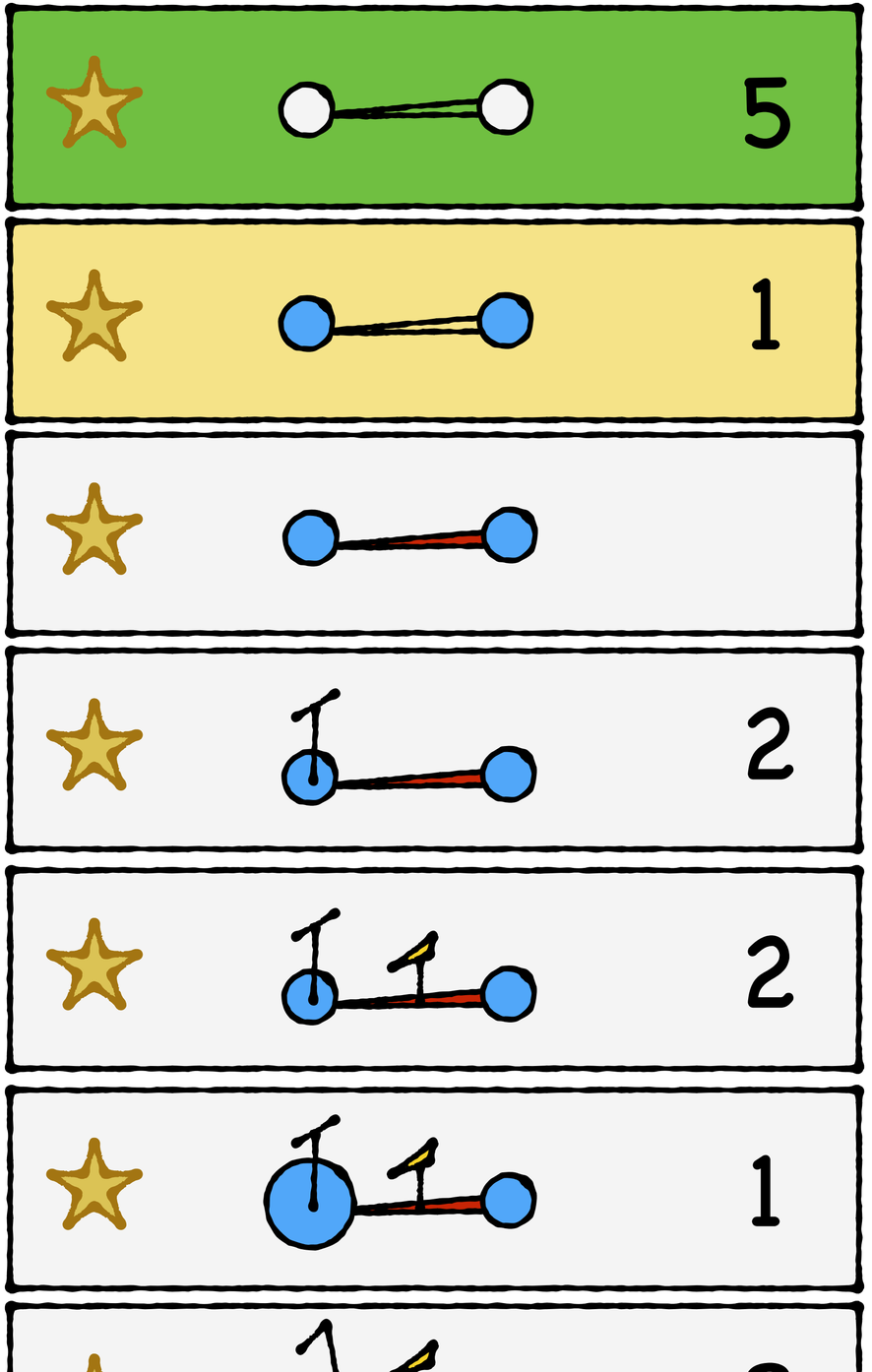




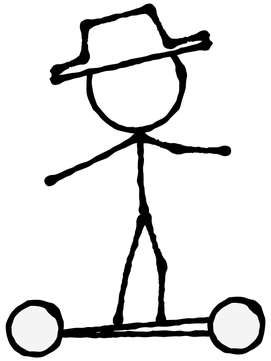
Short feedback cycles

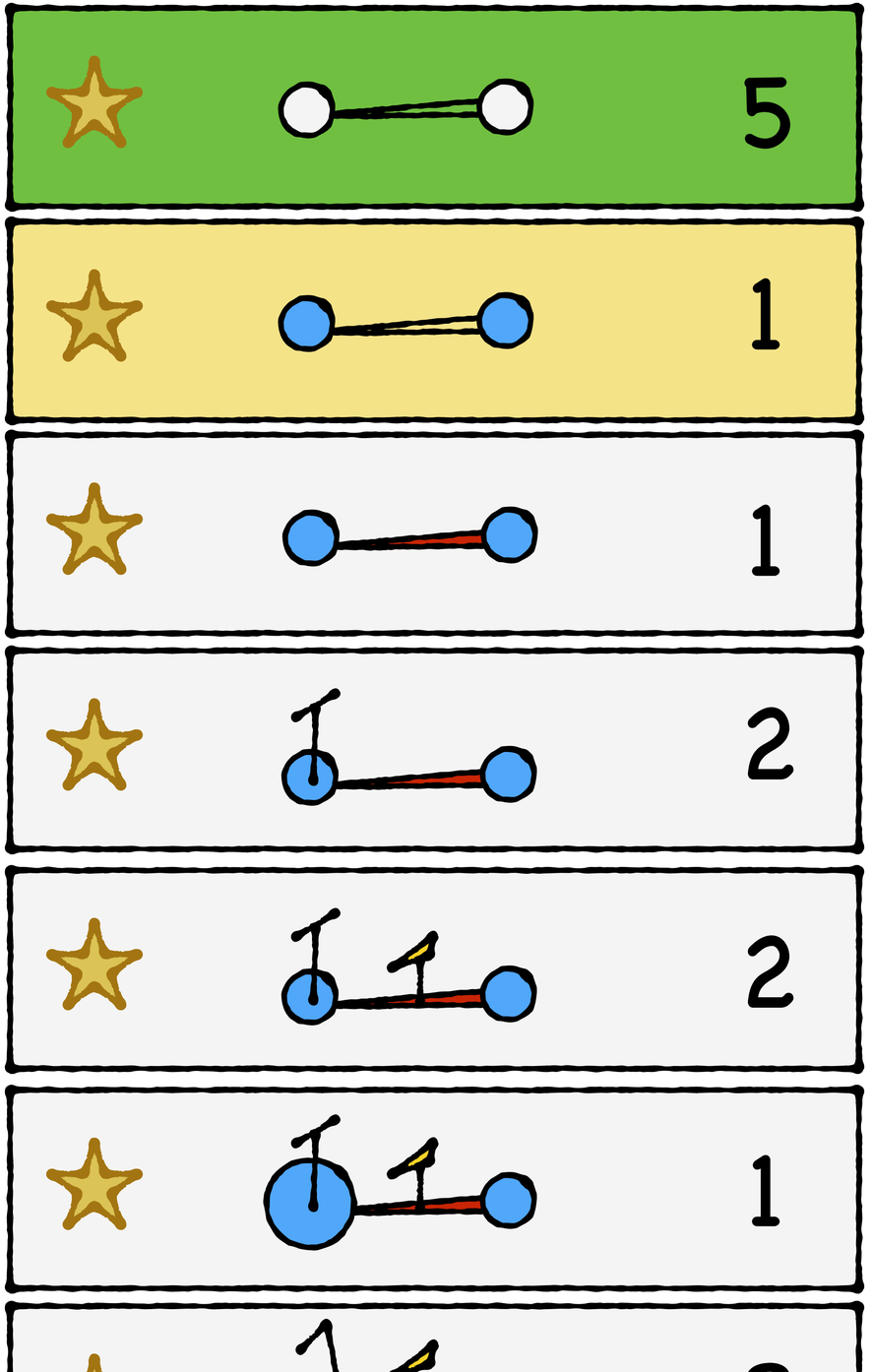






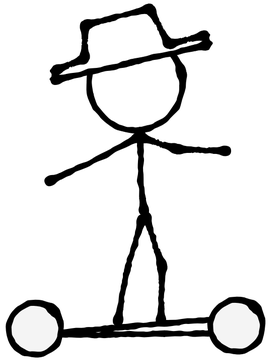
Short feedback cycles

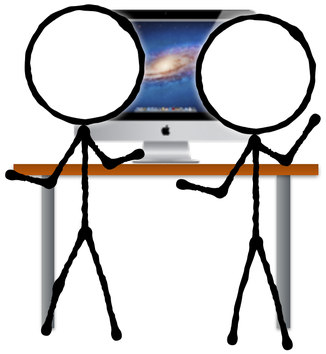


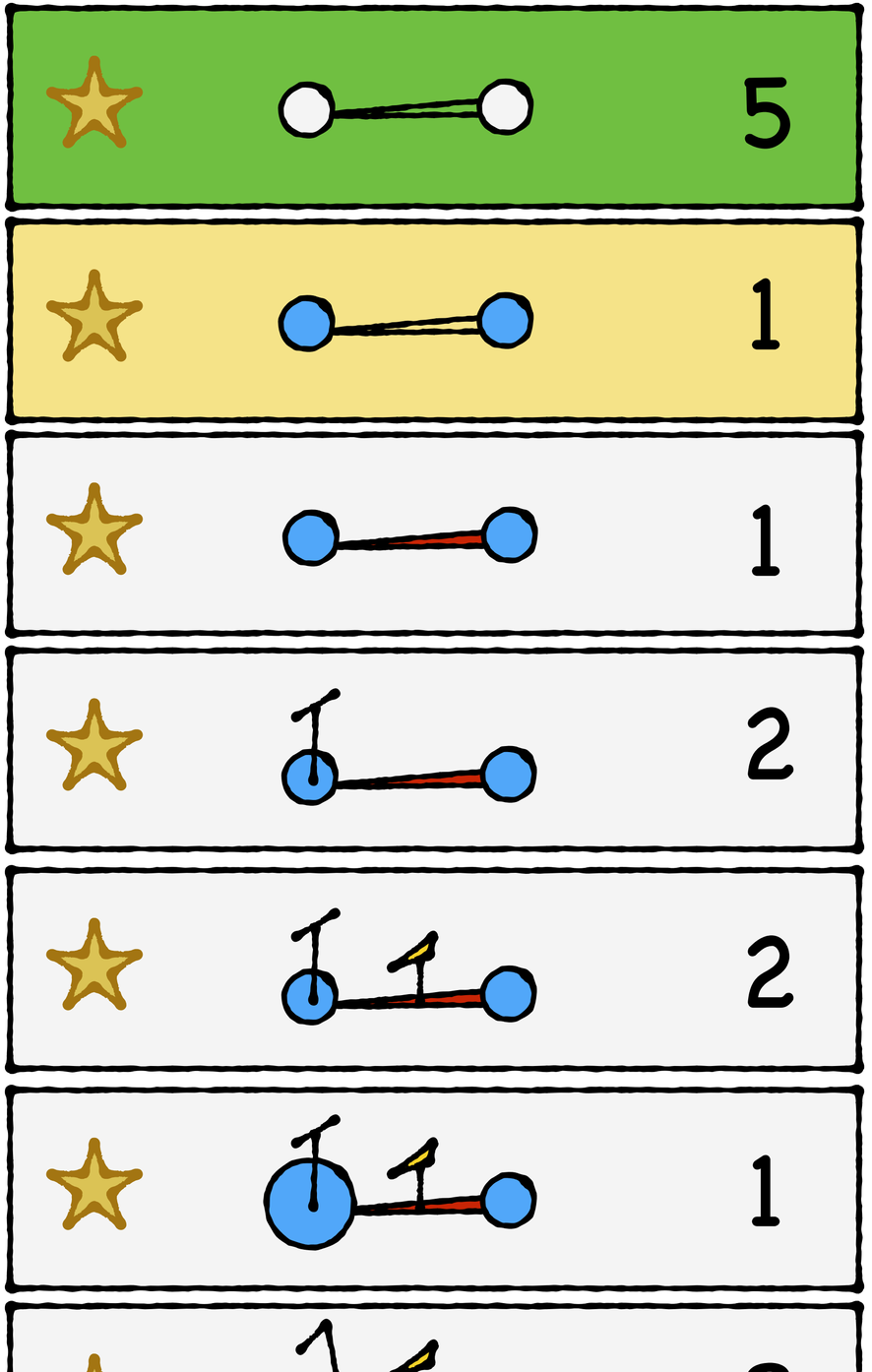


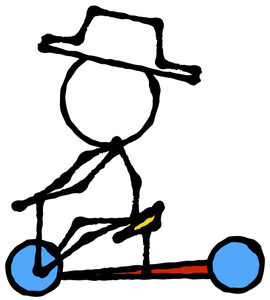
Short feedback cycles

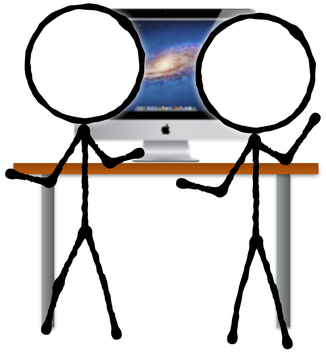
Changing design in response to feedback

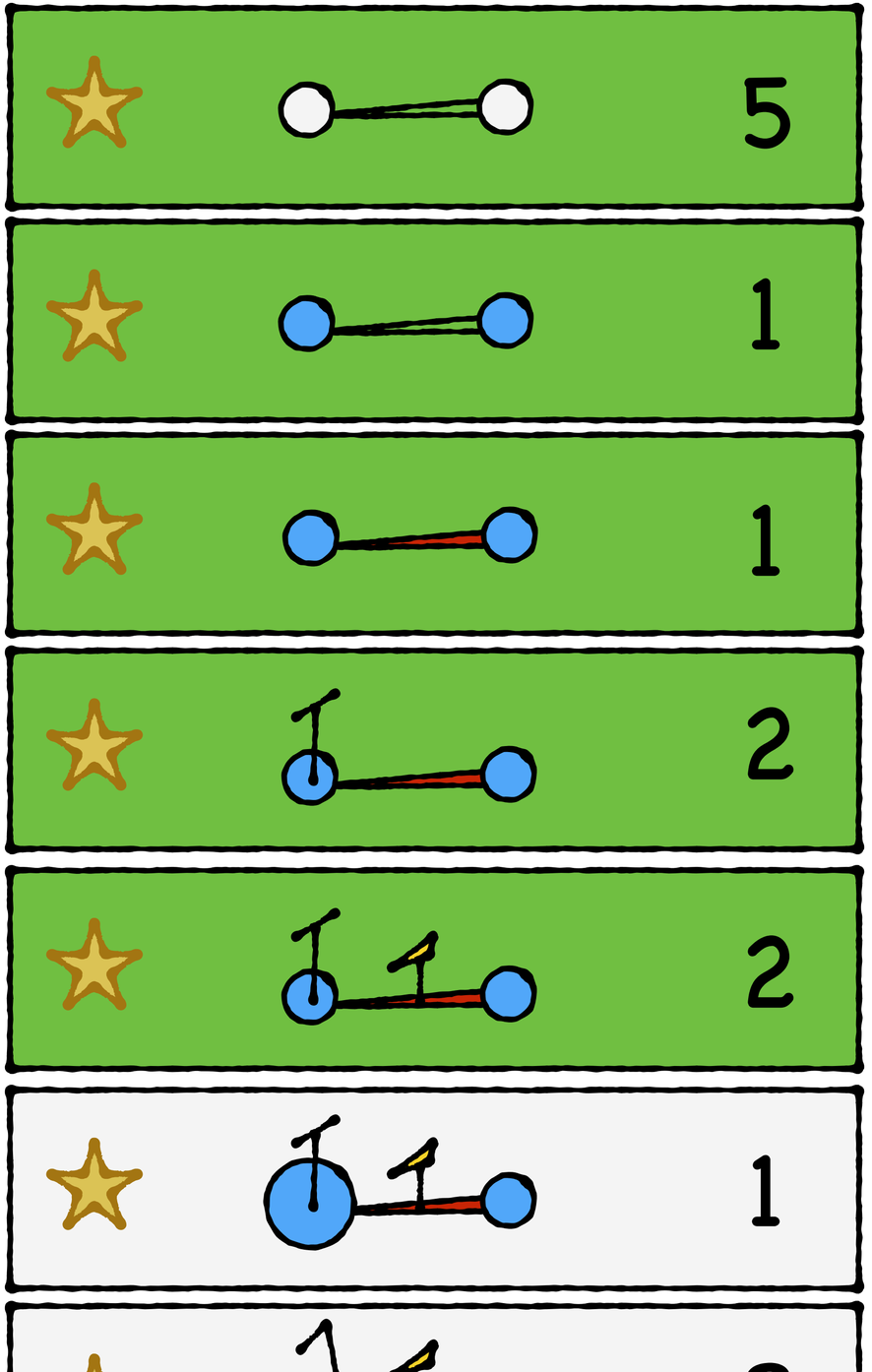






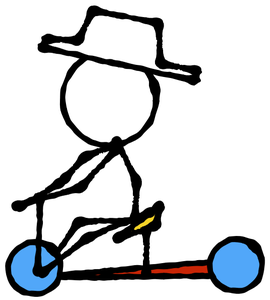


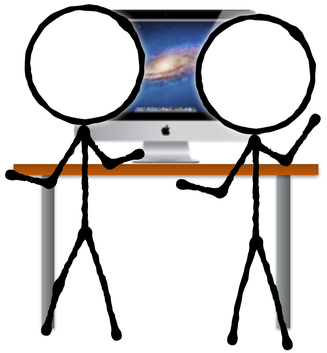


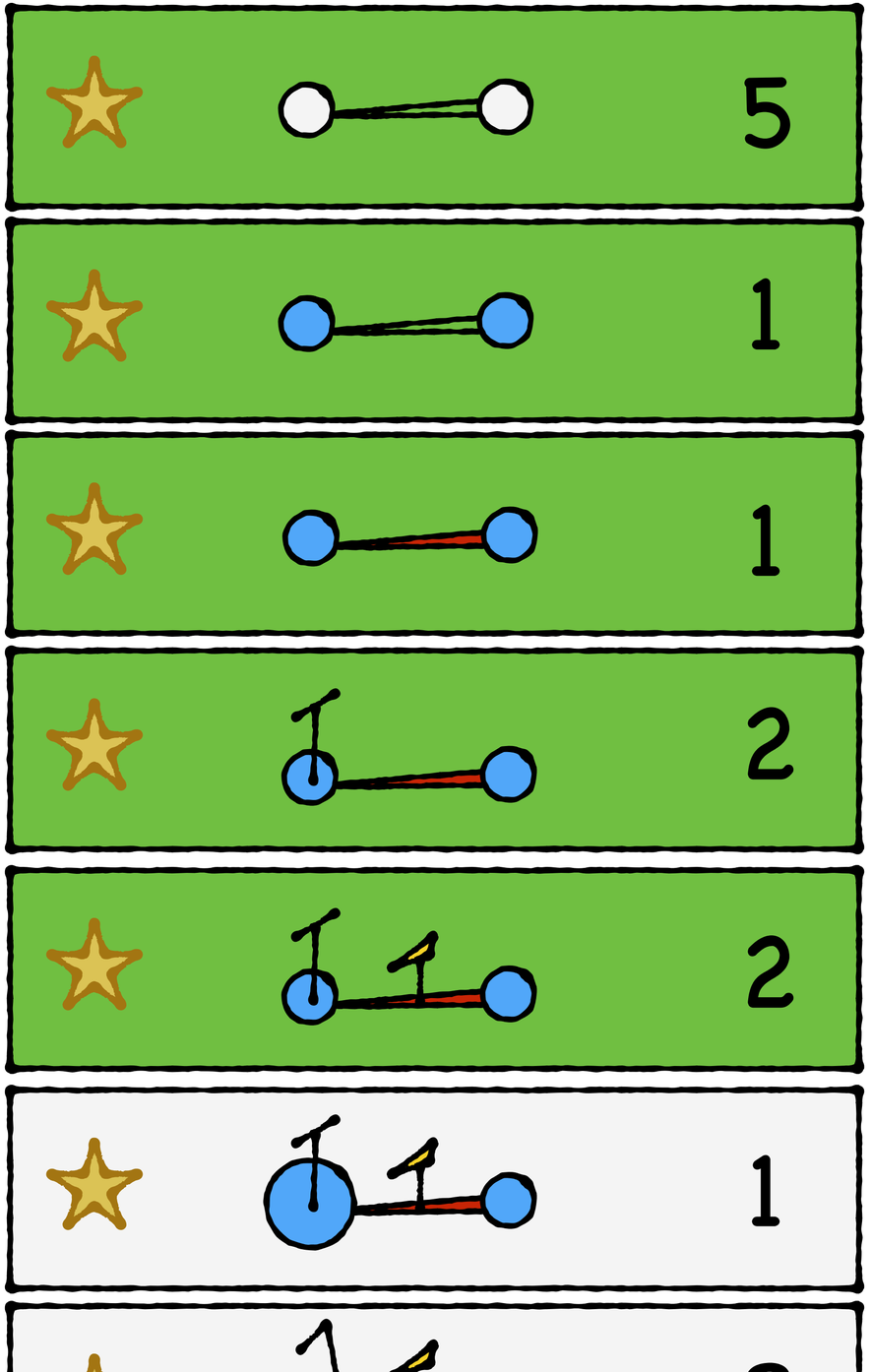


Short feedback cycles

Changing design in response to feedback



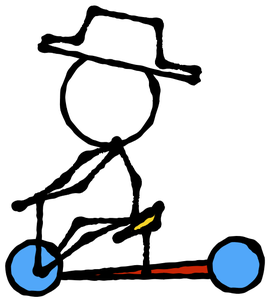


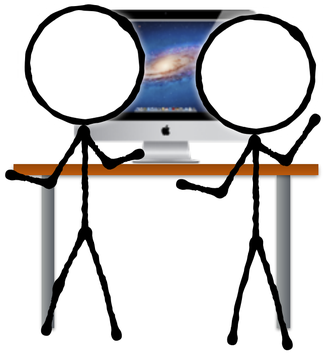


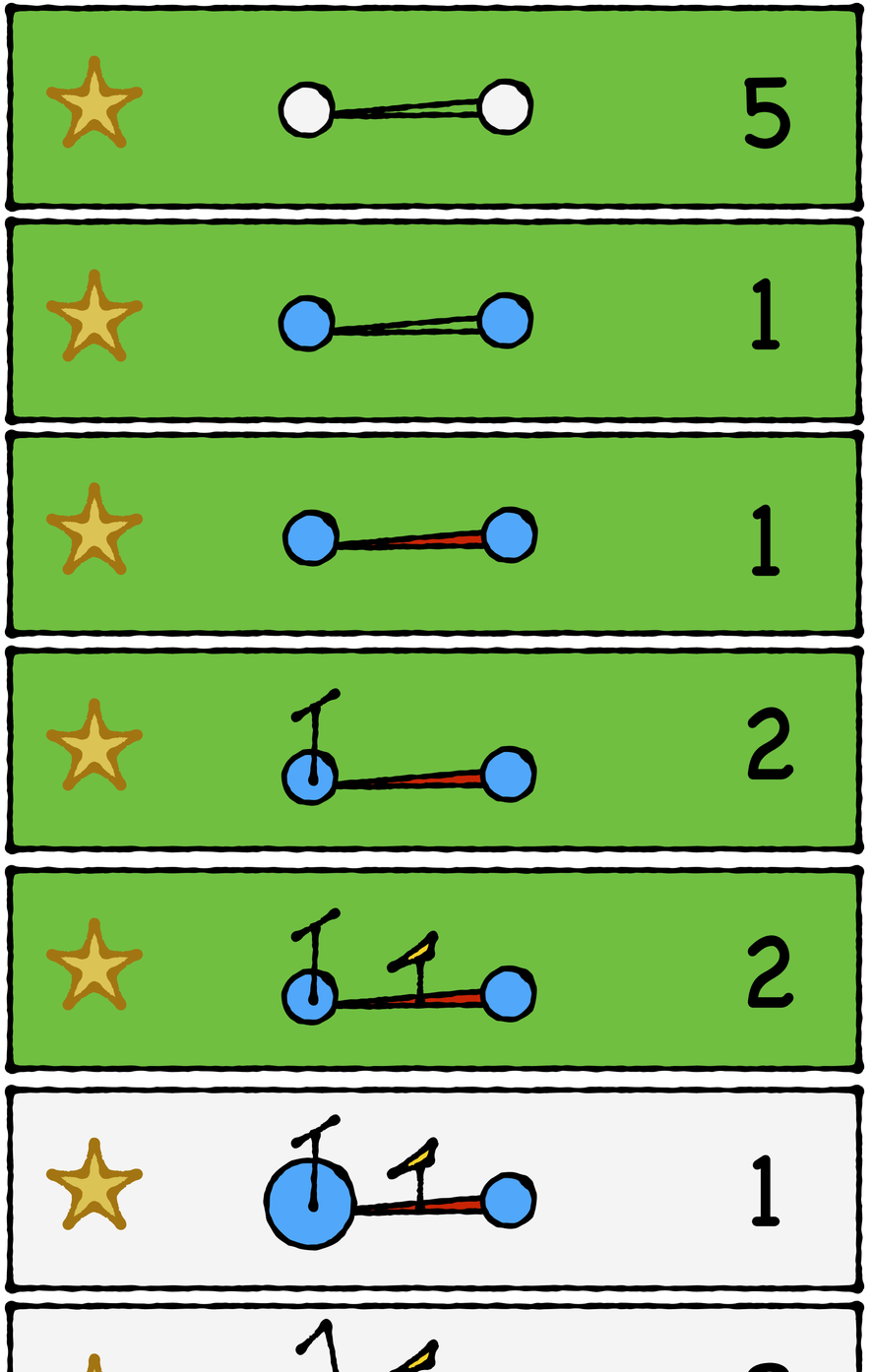
Short feedback cycles

Changing design in response to feedback

← DATA





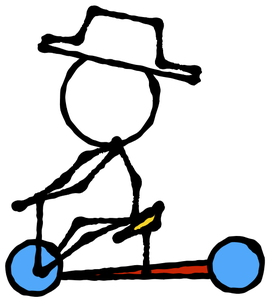


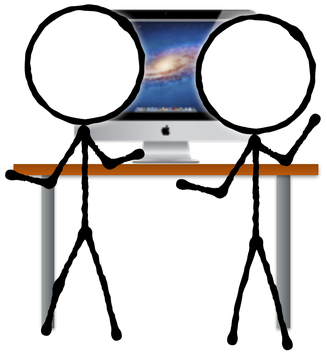
Short feedback cycles

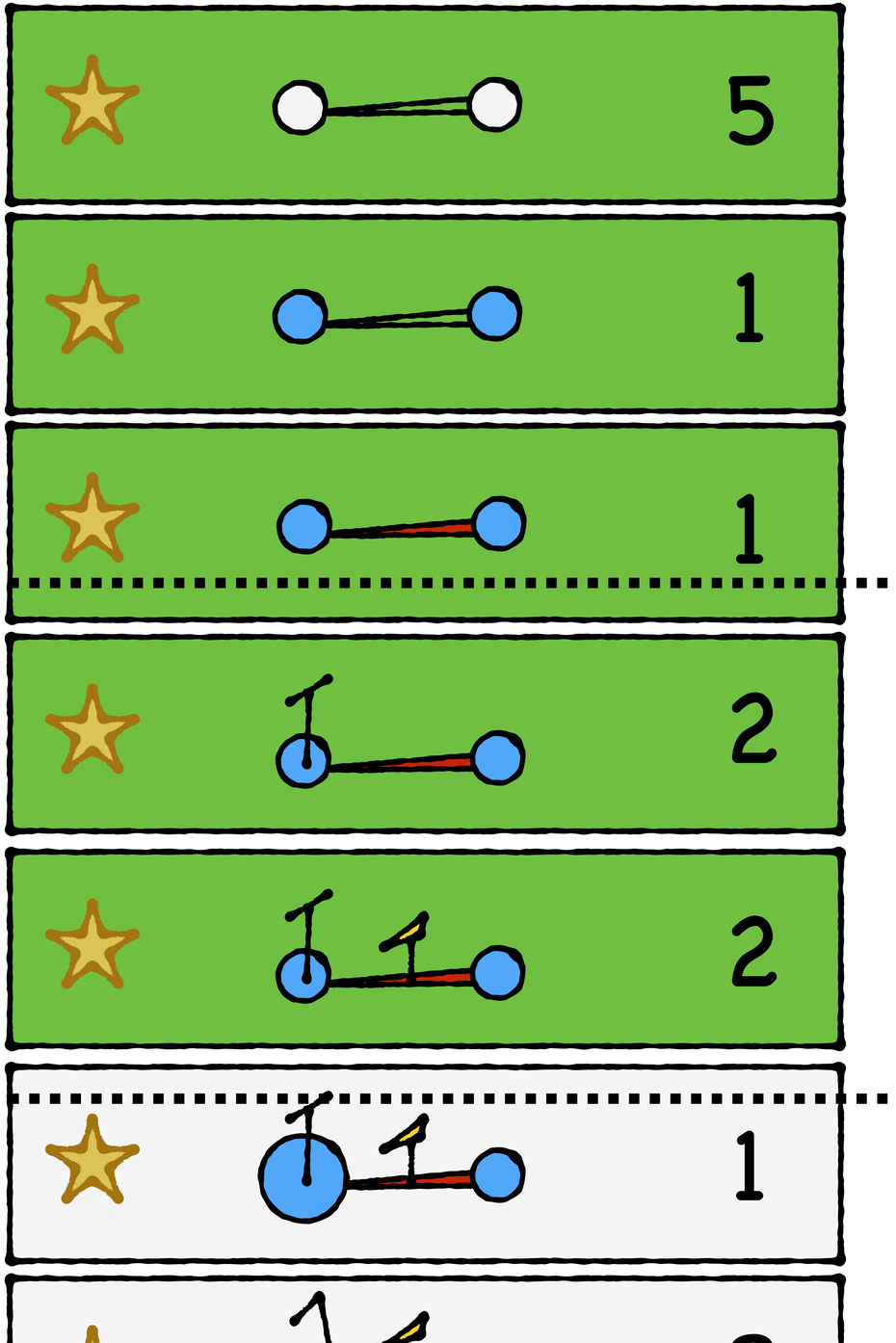
Changing design in response to feedback

← **DATA**

velocity = points/week







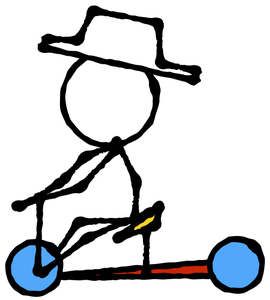
Short feedback cycles

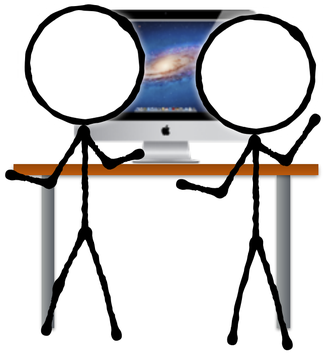
Changing design in response to feedback

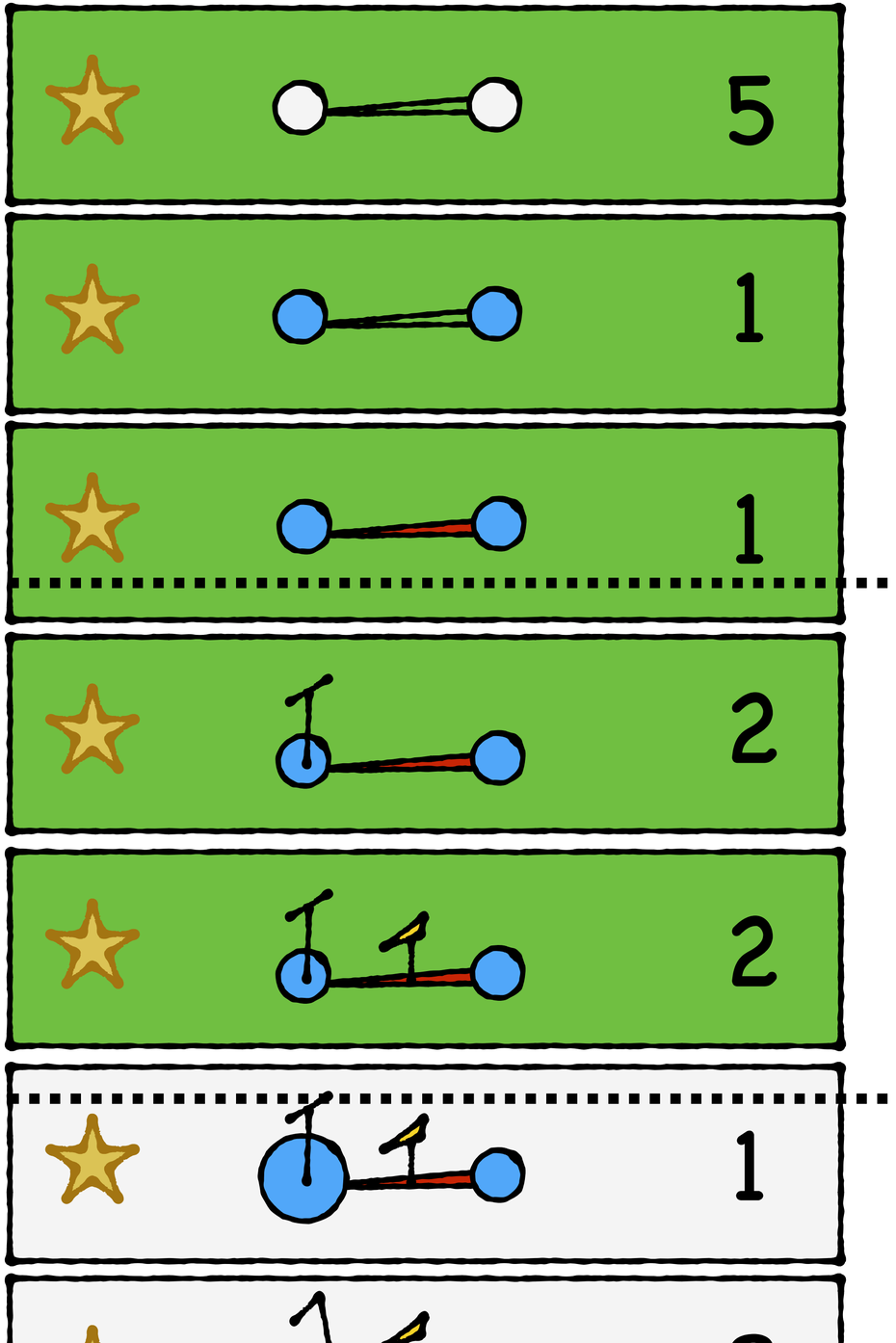
← **DATA**

velocity = points/week

Enables data-driven delivery predictions







Short feedback cycles

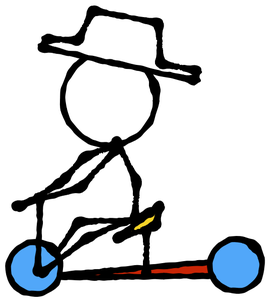
Changing design in response to feedback

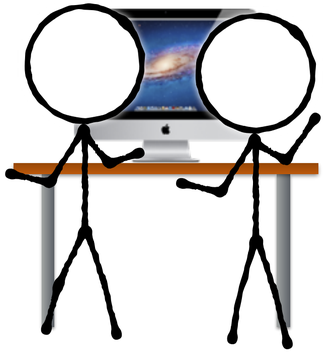
← **DATA**

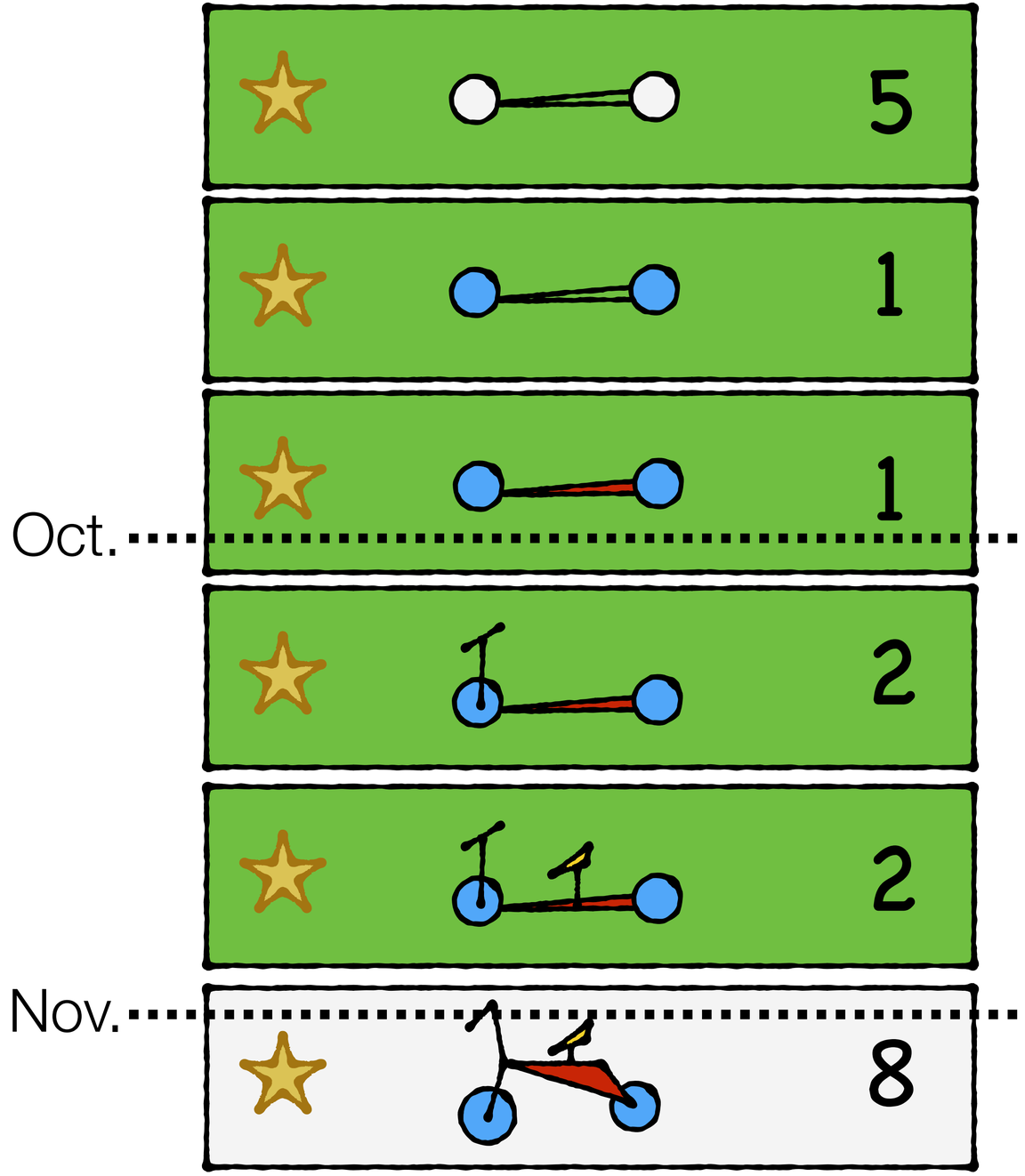
velocity = points/week

Enables data-driven delivery predictions

To hit a deadline:







Short feedback cycles

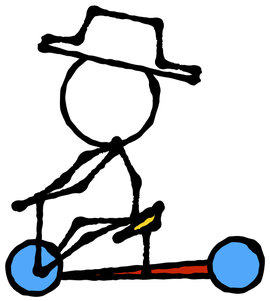
Changing design in response to feedback

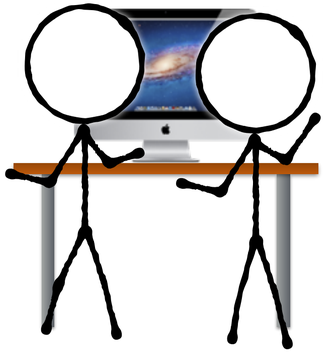
← **DATA**

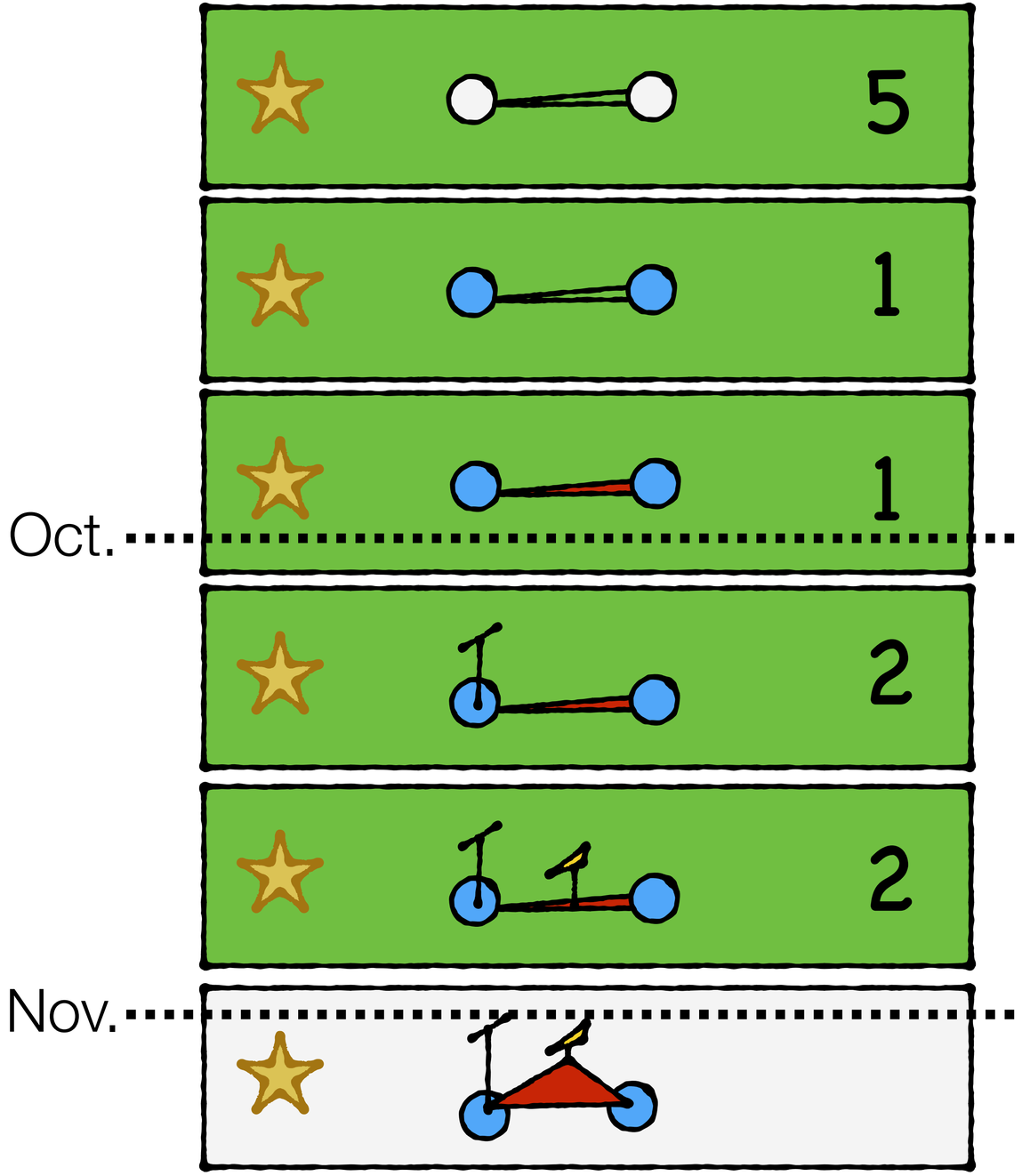
velocity = points/week

Enables data-driven delivery predictions

To hit a deadline:
Reprioritize







Short feedback cycles

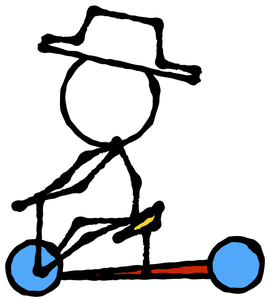
Changing design in response to feedback

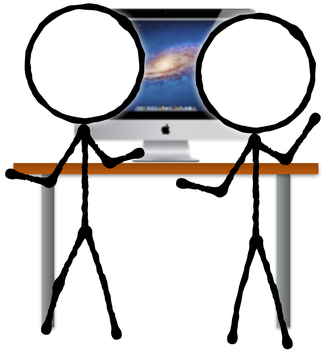
← **DATA**

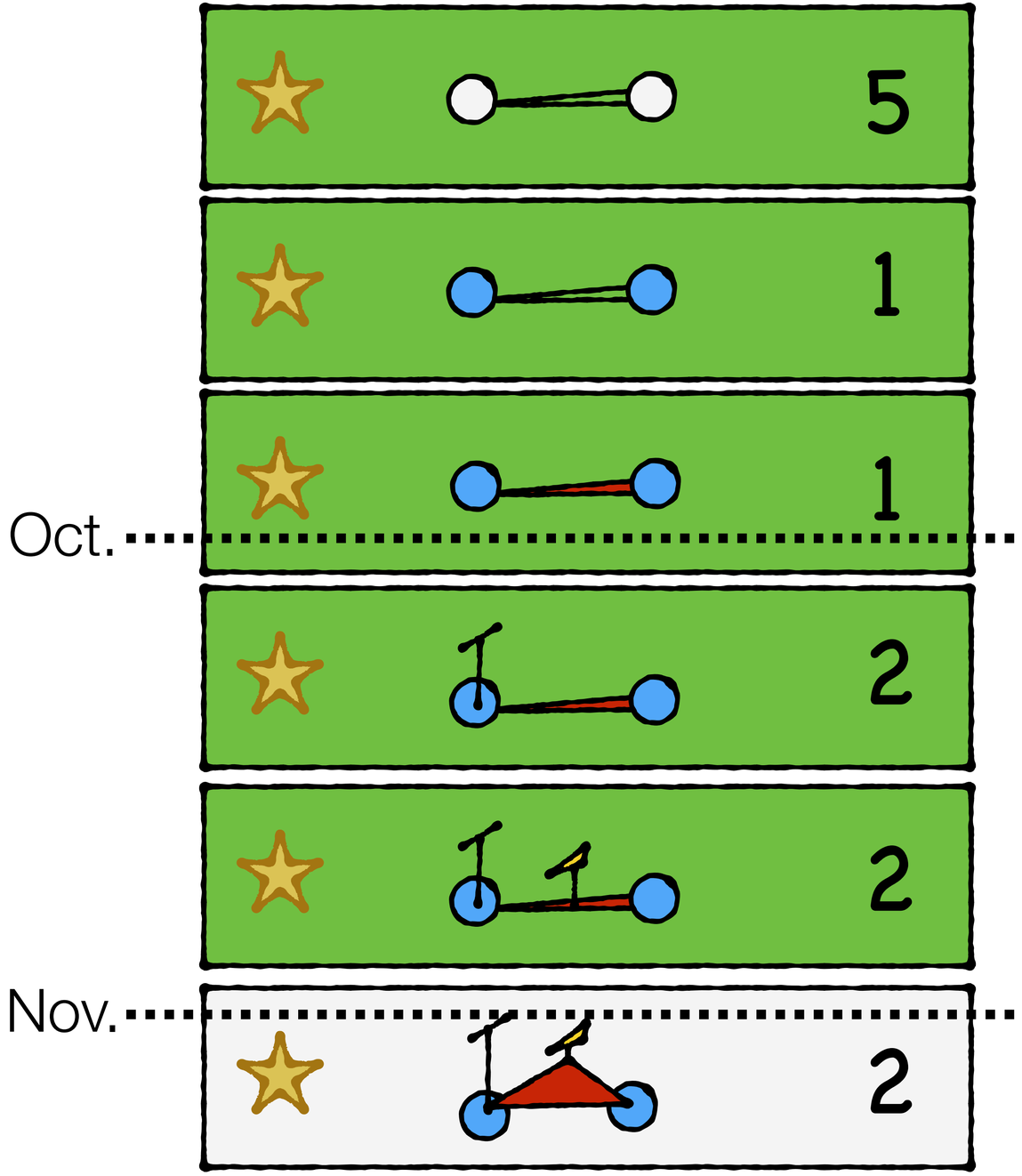
velocity = points/week

Enables data-driven delivery predictions

To hit a deadline:
Reprioritize
Reduce Scope







Short feedback cycles

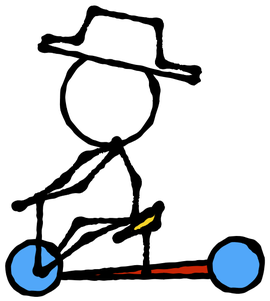
Changing design in response to feedback

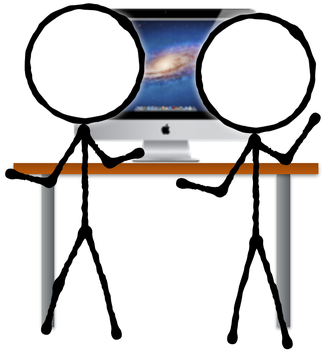
← **DATA**

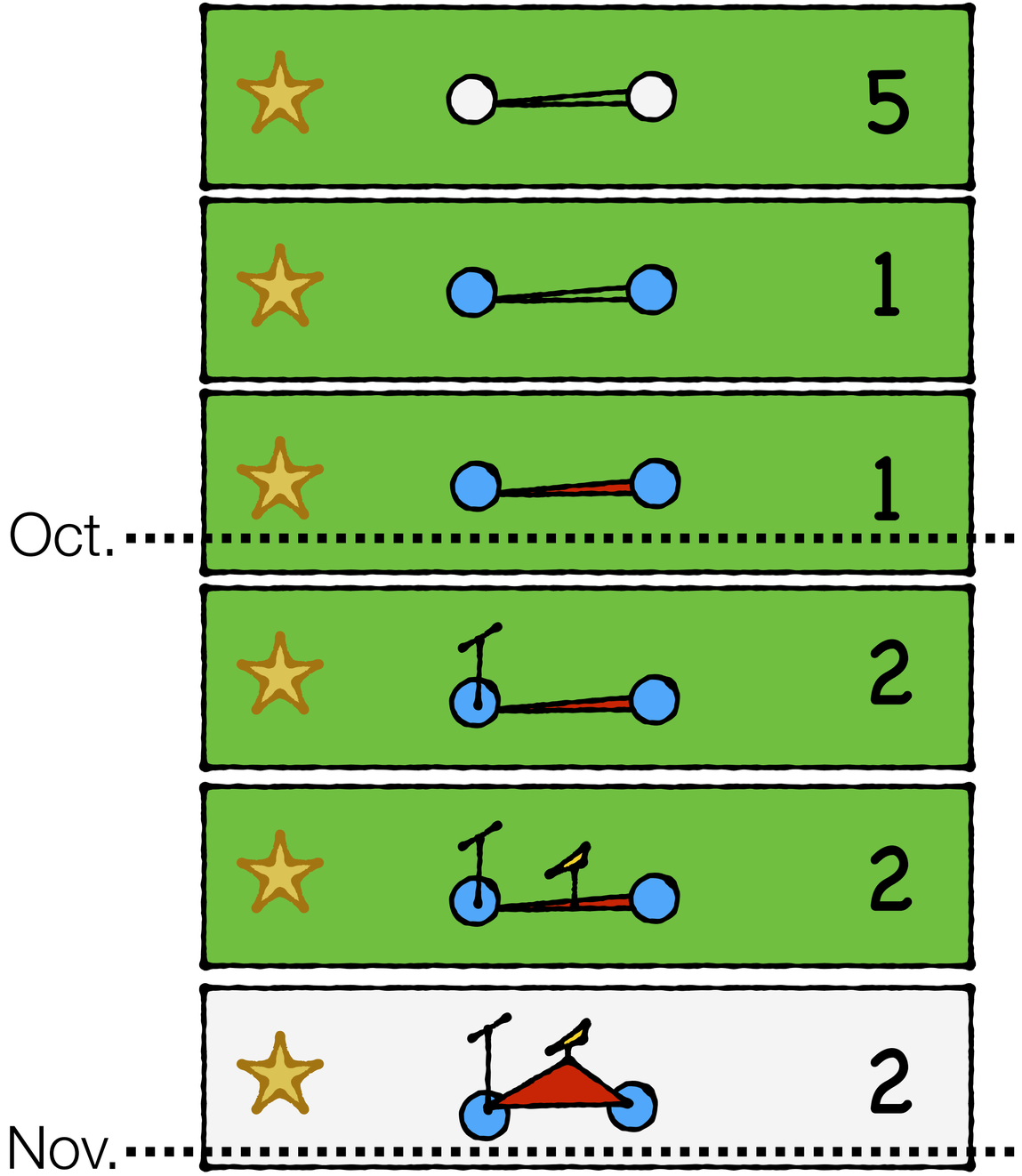
velocity = points/week

Enables data-driven delivery predictions

To hit a deadline:
Reprioritize
Reduce Scope







Short feedback cycles

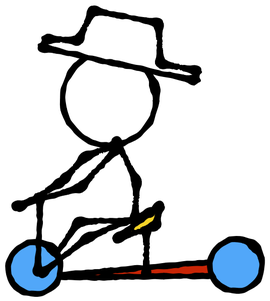
Changing design in response to feedback

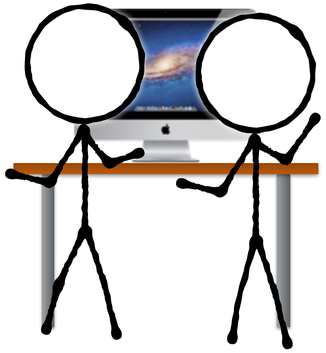
← **DATA**

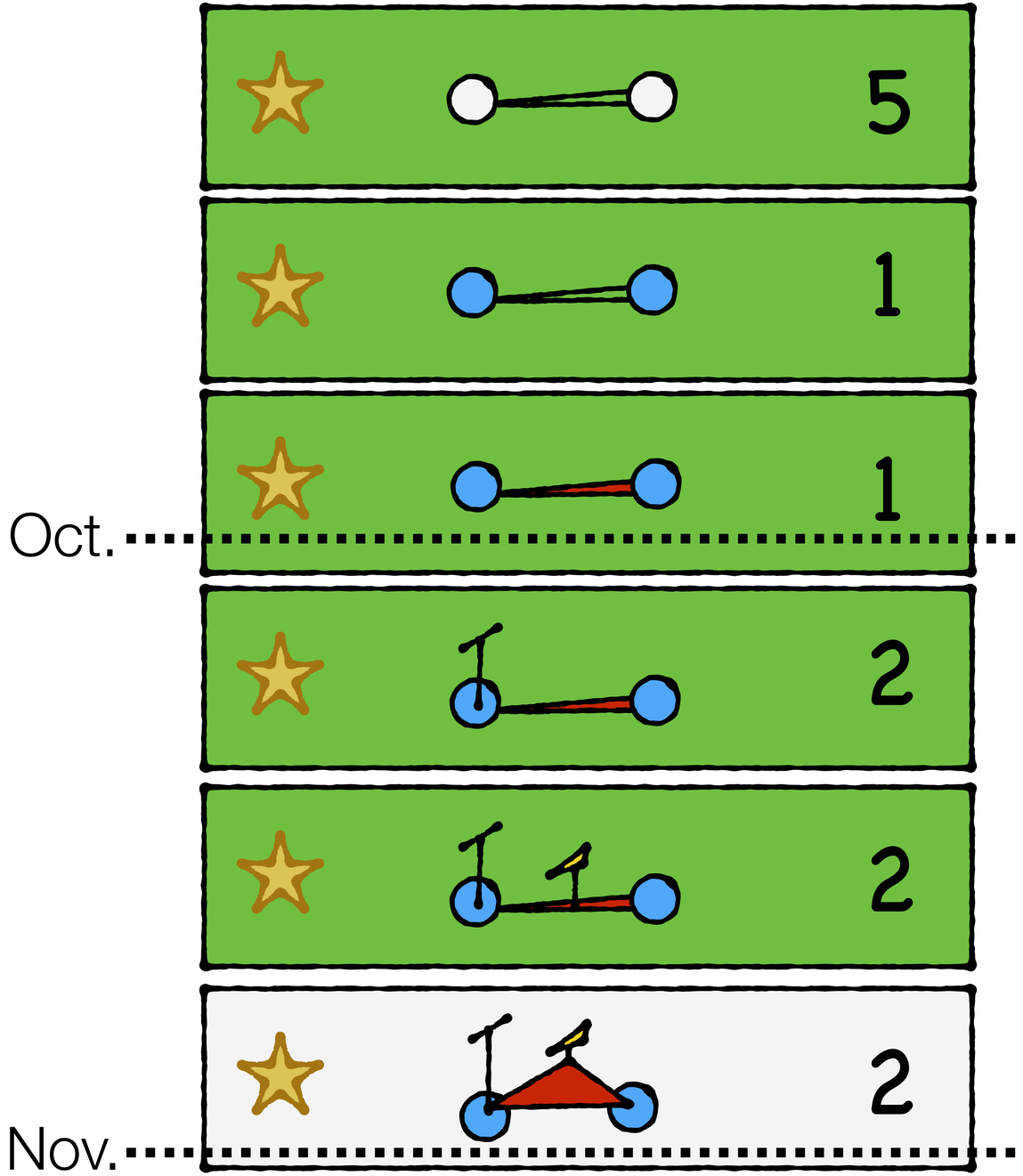
velocity = points/week

Enables data-driven delivery predictions

To hit a deadline:
Reprioritize
Reduce Scope



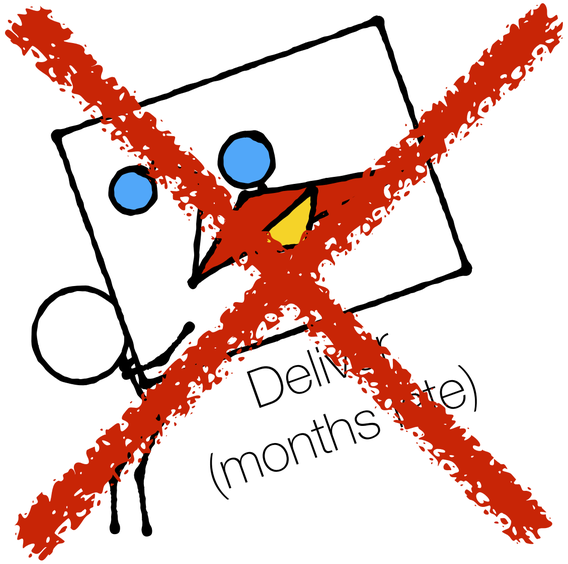




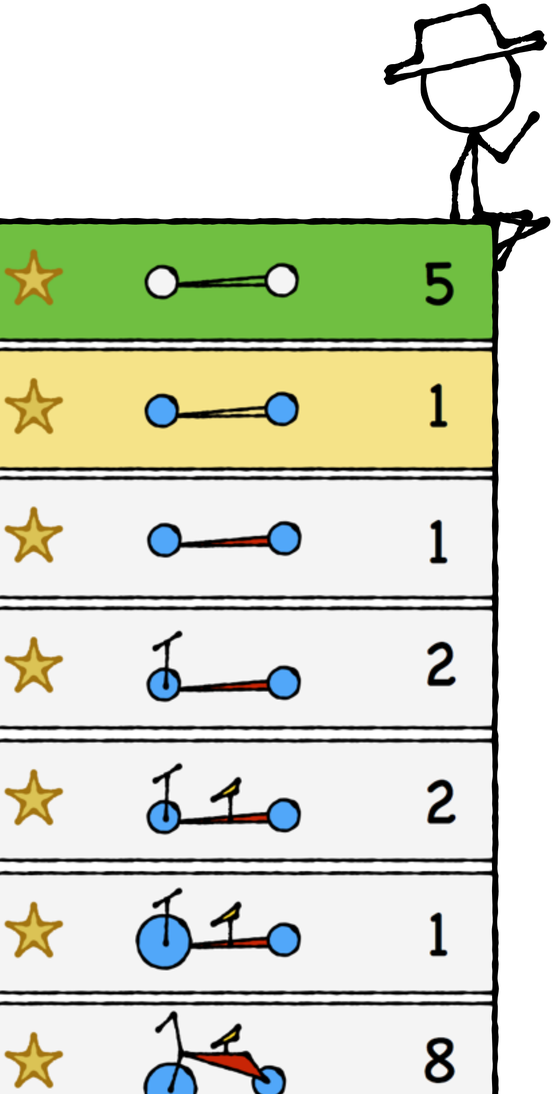
Short feedback cycles

Changing design in response to feedback

Data driven planning


Deliv
(months
te)

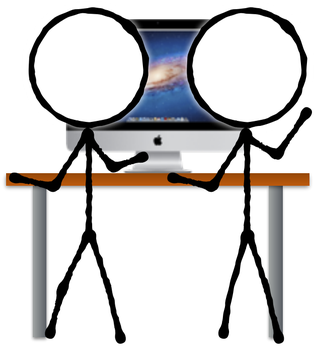
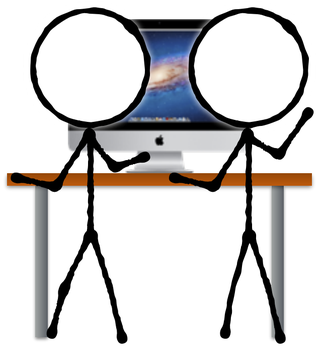
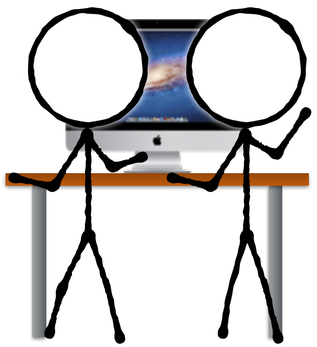

Stakeholders

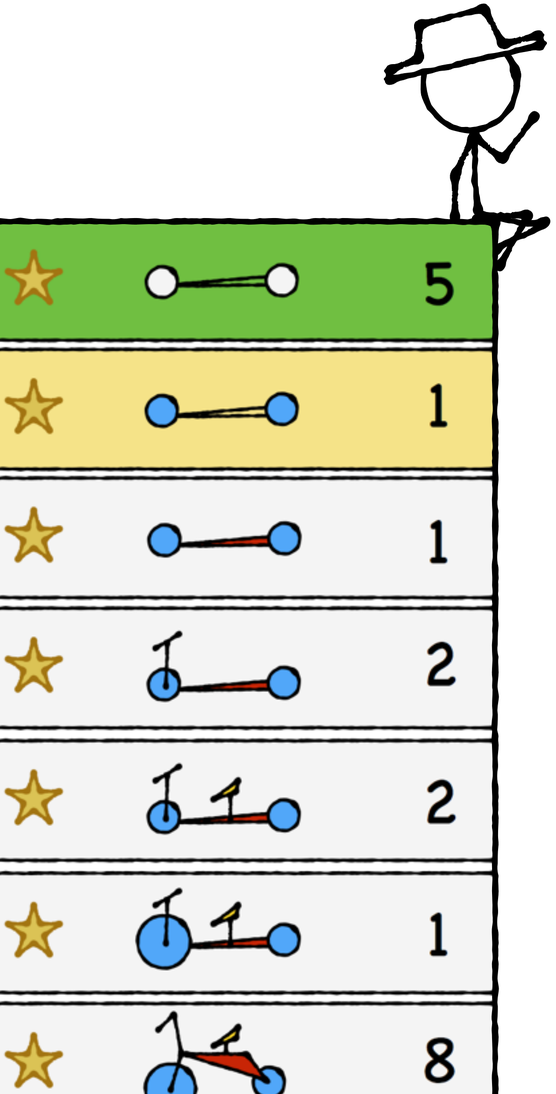


Pivotal™

culture

Why?



Pivotal™ culture

Humanizing

Building Clouds

http://pivotal.io

Pivotal™