# IBM GARAGE METHOD

# TODAY'S GOAL

- Overview of the XP process
- Project Inception

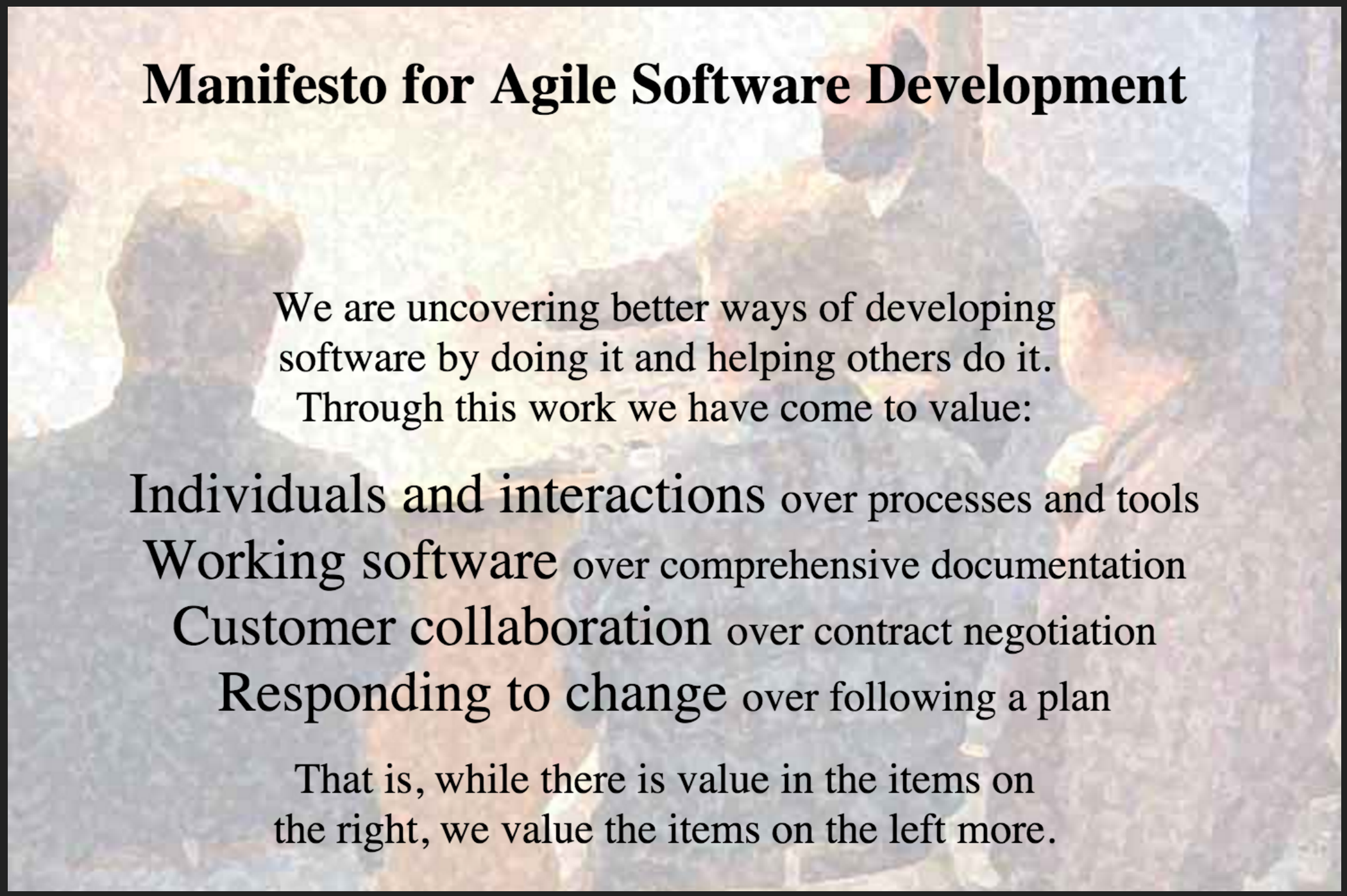
# AGENDA

- 9:00 - 10:30 Overview of Garage Method
- 10:45 - 11:00 Recap of Design Work & MVP
- 11:30 - 12:00 Goals and Non-Goals/Risks
- 12:00 - 1:00 Lunch
- 1:00 - 1:30 Users/User activities
- 1:30 - 2:00 Overview of writing stories and story mapping
- 2:00 - 4:00 Story mapping – Writing Epics and Stories
- 4:00 - 5:00 Retrospective

# WORKING AGREEMENTS

- Be open to possibilities
- Minimize side conversations
- Keep track of the time
- Full hour for lunch – breaks
- Be fun/have some fun

# THE AGILE MANIFESTO

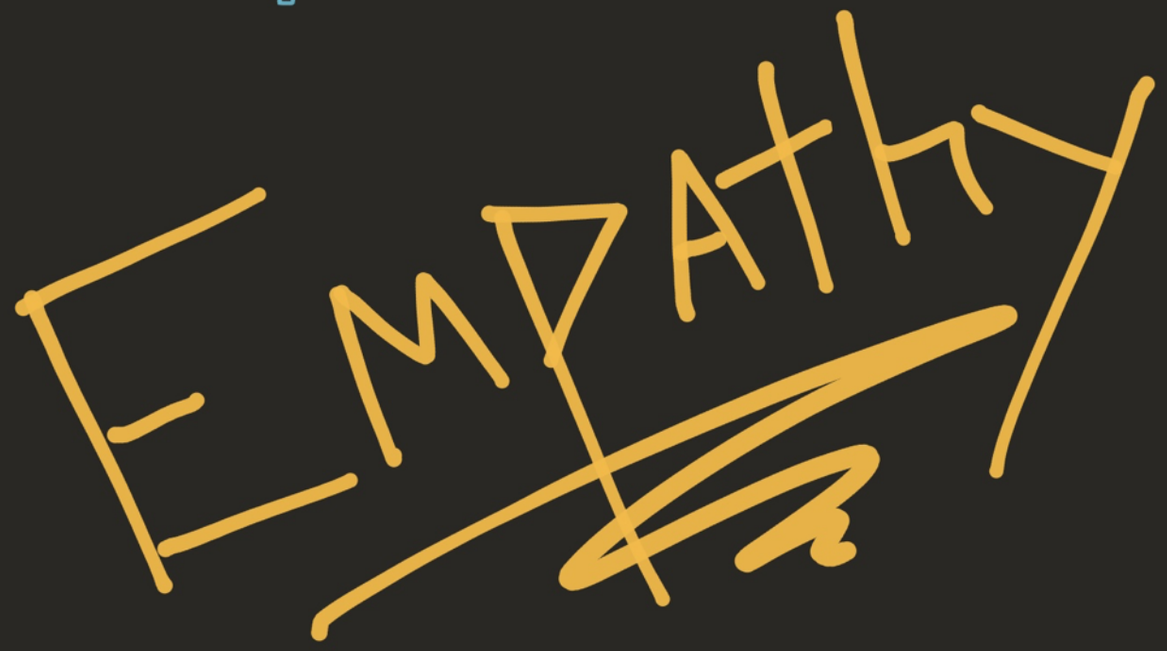


# XP

- Value based flavor of agile
- The idea is that we do more of the things that work and less of the things that don’t

# VALUES

Empowerment




Trust

Transparency

# THE XP TOOLKIT

- Pair Programming
- Co-location
- Test Driven Development – TDD
- Continuous Integration/Continuous Delivery
- Small User Stories
- Velocity
- Weekly Iterations

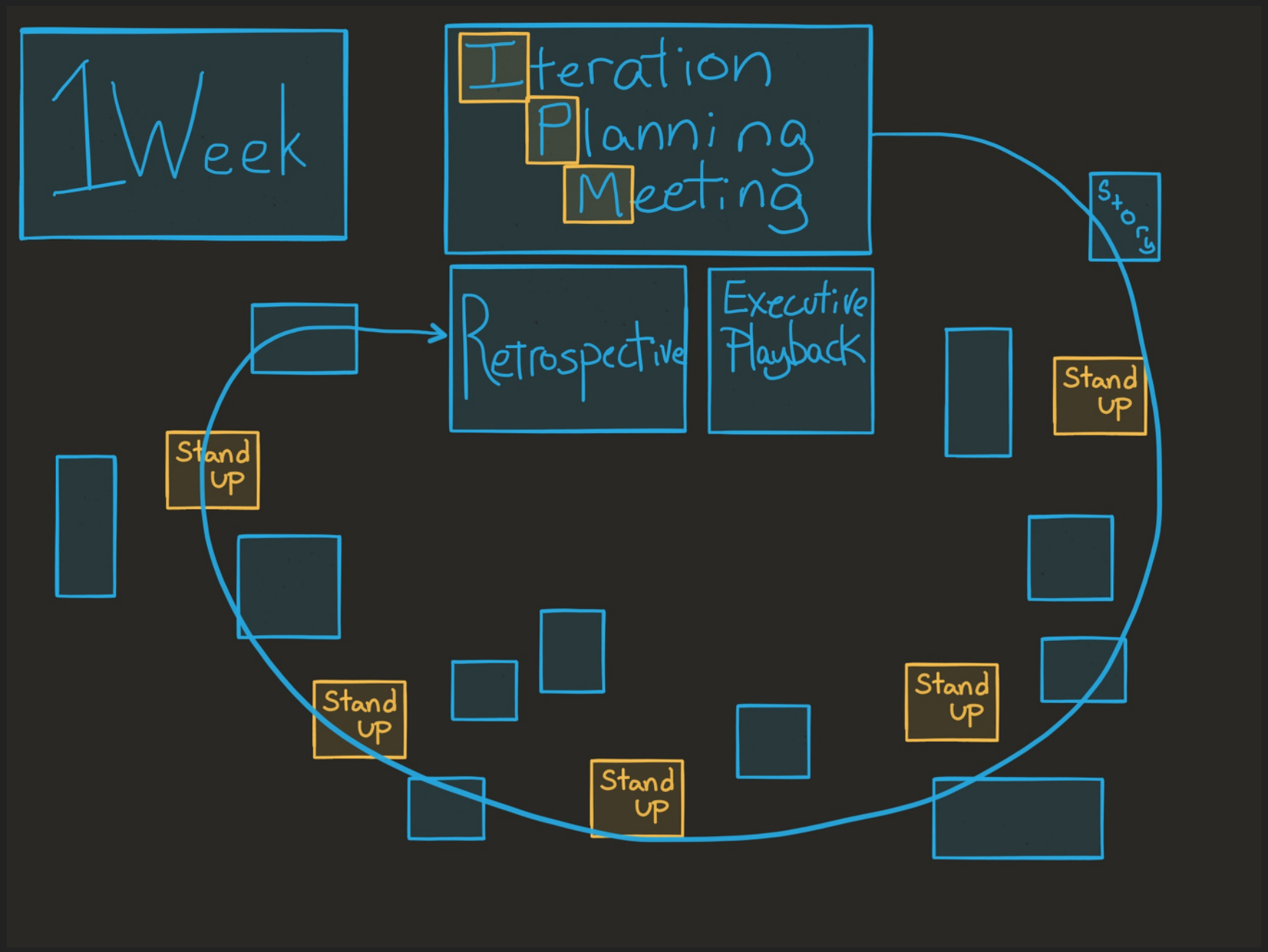
# XP

- 1 week iteration
- Stories pulled from continuous backlog
- Stories assigned when pulled
- There is always an iteration

# Scrum

- 2-4 week iteration
- Committment made to stories at iteration Beginning
- developers assigned stories when iteration Begins

# WEEKLY PROJECT FLOW



# ROLES



# ROLES IN THE PROJECT FLOW

- Runs the IPM
- Answers Questions
- Accepts stories as they are completed
- Answers more Questions
- Looks at analytics

# ROLES IN THE PROJECT FLOW

- Validate MVP
- Work with developers on bringing wireframes to life
- User testing
- Iterate on the design

# ROLES IN THE PROJECT FLOW


- Work on stories in pairs
- Test drive
- Ask PM and Designer questions
- Might have an affinity in front-end, back-end or dev ops but work together to get things done…this is full-stack engineering


Stories

Written From the perspective of the User

Every Story delivers Value

End to end

Emma sees her profile

# USER STORIES

- the smallest unit of functionality that delivers value to the user
- good stories should be able to be completed in less than a day
- developers will assign stories point based on complexity – these are not time estimates

# STORIES ARE SMALL AND BUILD ON EACH OTHER

1 Emma sees a page

2 Emma sees Profile inFormation

3 Emma logs in

Working Software

Building up in small pieces

End to end Functionality

# TYPES OF STORIES

- Features – Deliver an outcome
- Chores – Technical debt
- Bugs – Also technical debt

Emma sees a login

Page 2

Done

# POINTS

Done

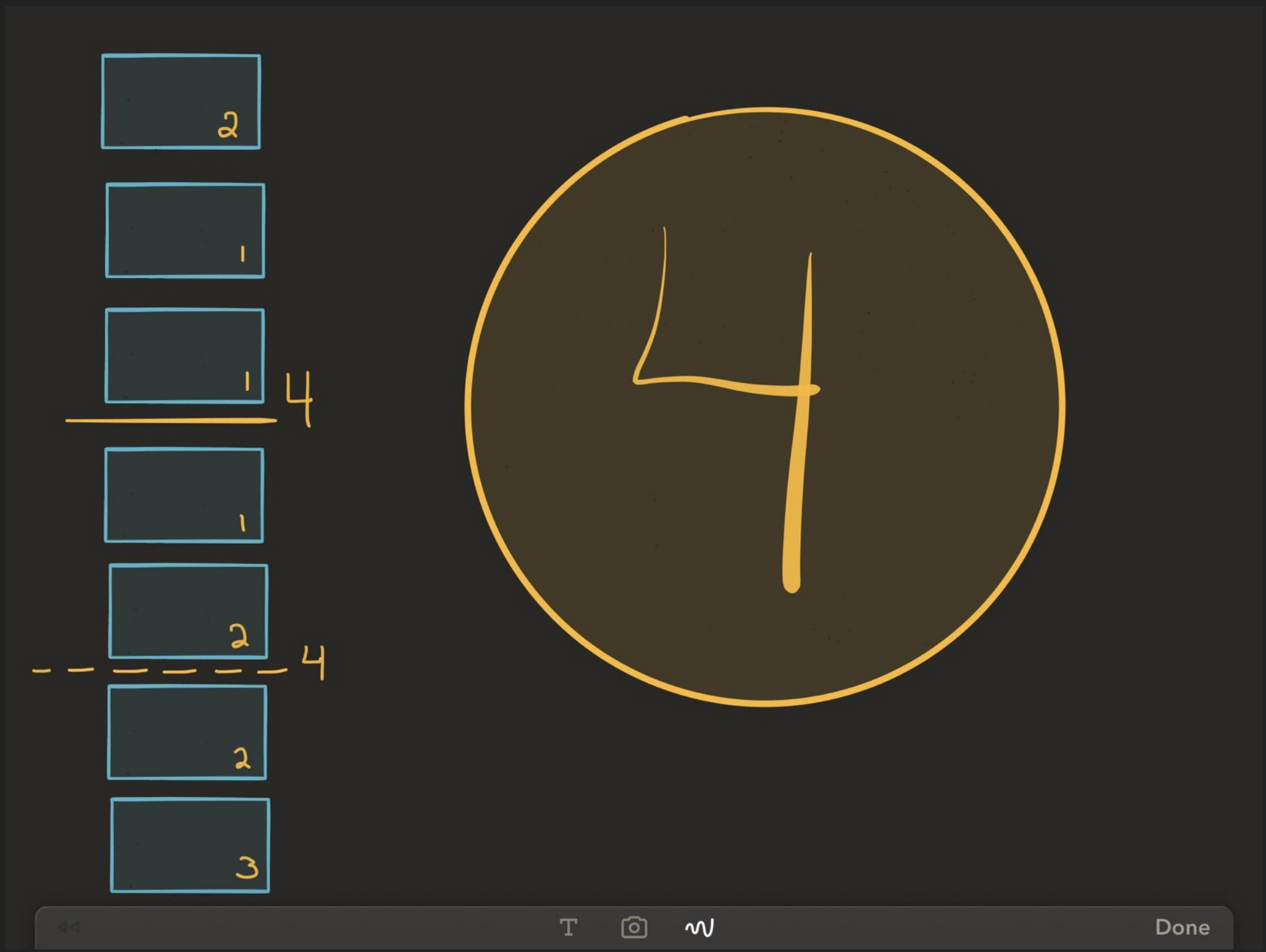
# PLANNING POKER

- Talk about the story – what is involved to make it happen? What is the complexity?
- Someone counts to 3
- Devs hold up fingers corresponding to number of points
- 1, 2, 3, 5, 8
- If you disagree…talk about it and do it again
- If it's really big, maybe break the story down
- Remember, these are BAD GUESSES

# VELOCITY

- a predictive measure
- total story points per week

# VELOCITY



Someday

Backlog

Current

Done

Prioritized

what the team is doing today

# EPICS

# SUGGESTED TOOLS

- sprint.ly

# PLANNING!

- Inception
- Iteration Planning Meeting

| Inception | Iteration Planning Meeting |
| --- | --- |
| Project Kickoff | Once a Week |
| Goals + Risks | |
| Breakdown Stories | Breakdown Stories |
| Shared Understanding | Shared Understanding |
| | Estimate Points |

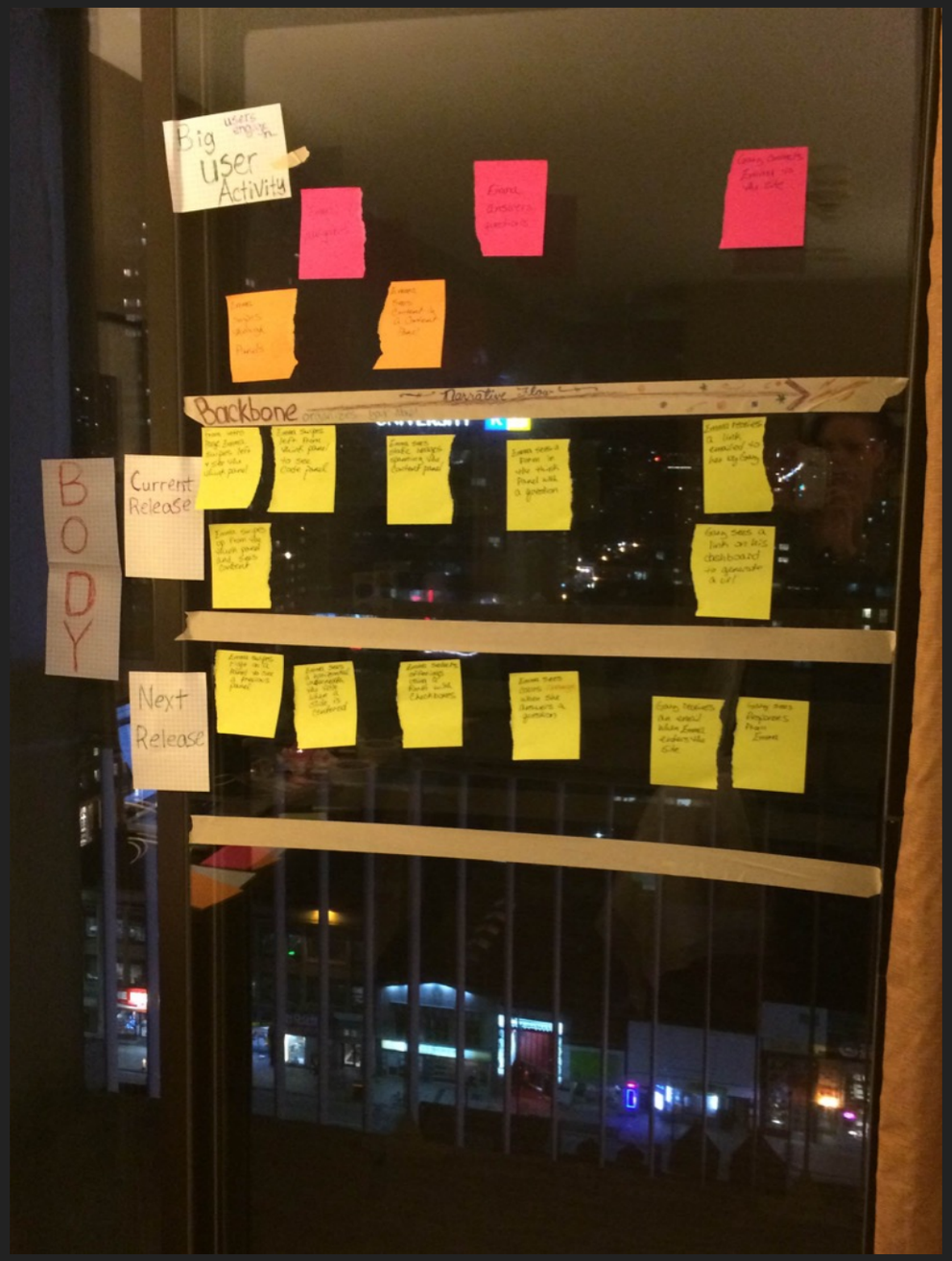
# INCEPTION

- Goals vs. non-goals
- Risks
- User Roles – Who did we miss? Who is lurking?
- Roles and activities – what are the major flows that deliver value?
- User Story Mapping – breakdown and prioritize stories
- Estimate the first iteration

# INCEPTION GOALS

- Build a backlog for the next 1 - 2 iterations
- Get an idea of what is reasonable for business

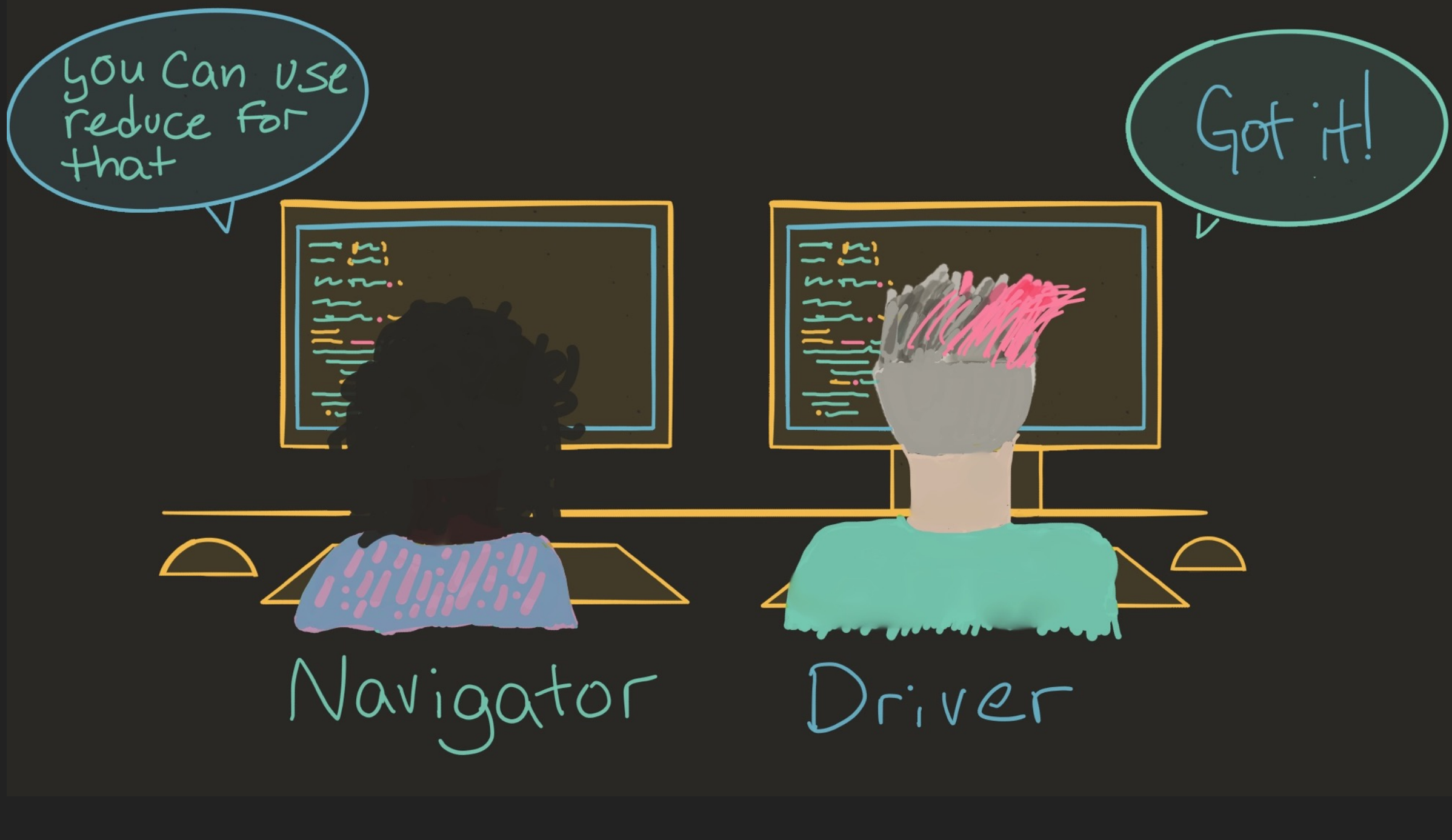
# USER STORY MAPPING



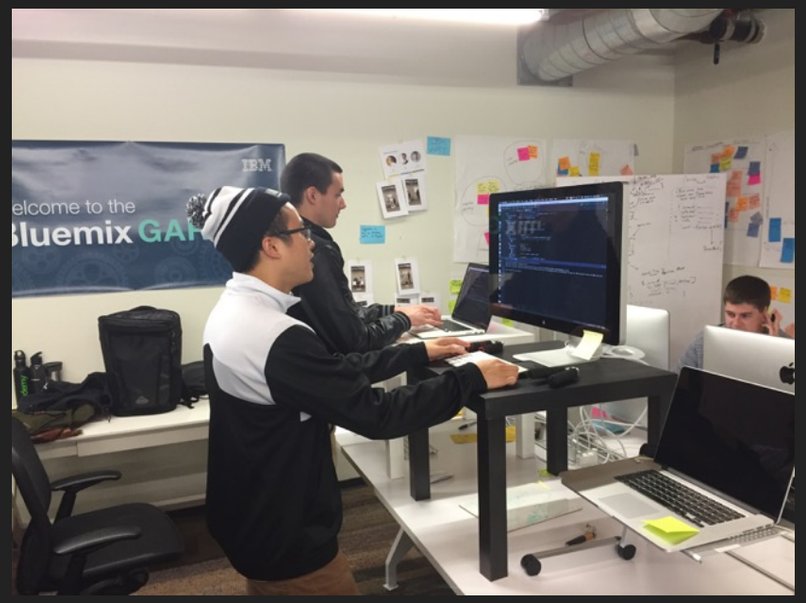
# XP DEVELOPER PRACTICES

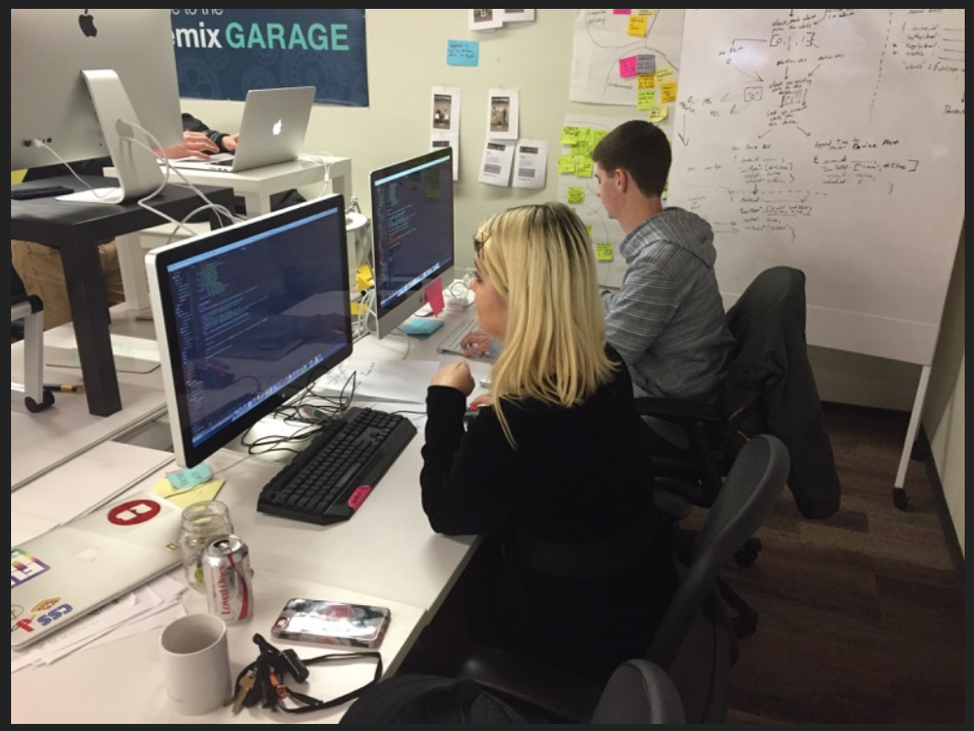
- Pairs
- Continuous Integration
- Test Driven Development
- Let’s Do this!

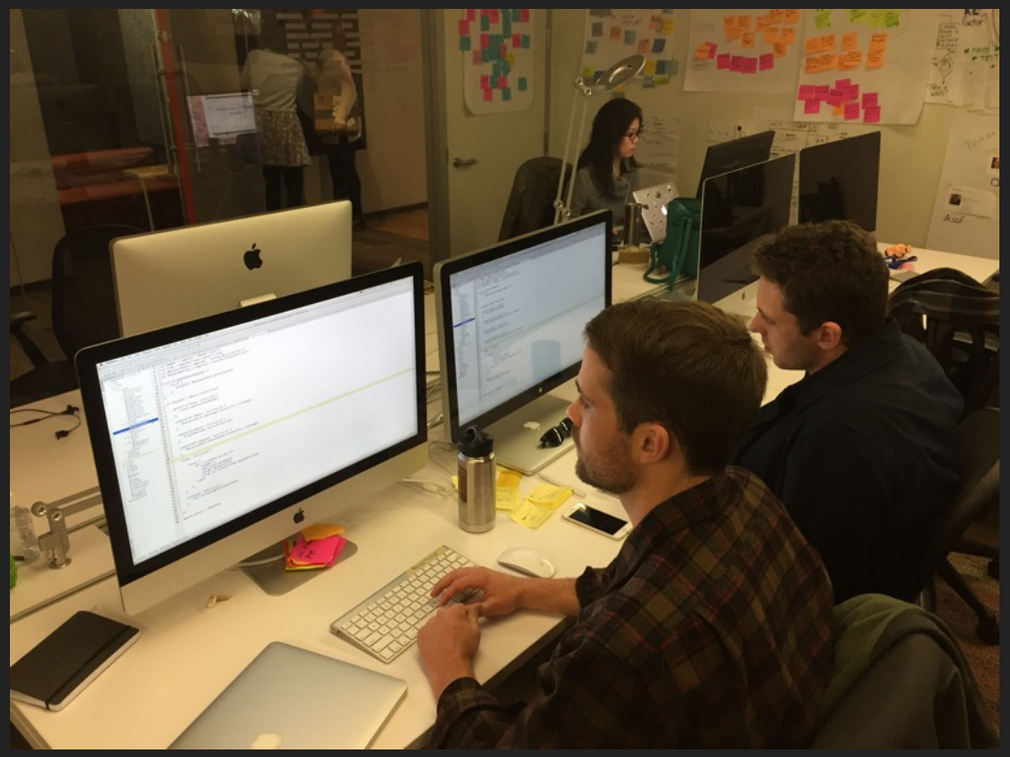
# PAIR PROGRAMMING



# PAIR PROGRAMMING



# HOW DO YOU PAIR?

- Timed intervals of driving 5 minutes, 10 minutes, 30 minutes
- It is ok to feel awkward about it
- Decide who is driver and who is navigator (often it is better if the less experienced coder drives)
- Pairs rotate daily

# A Day in the life of a pair

9:15

What's the story?

Standup!

Who is pairing?

Pair

Pair

Pair

12:30-1:30

## Lunch

Pair

Pair

Pair

## Ping Pong!

6:00

Commit →

tests Pass!

Dev

Staging

Production 1

Production 2

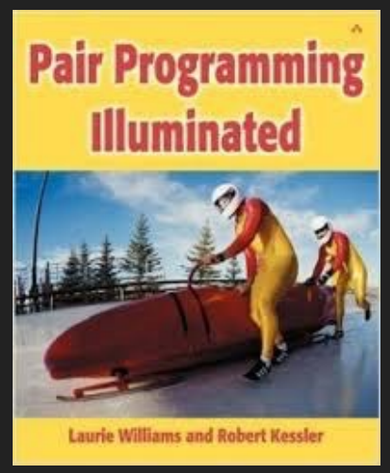
Commit

Failing tests (try it again)

Done

# REALLY?. . . YES, REALLY.

- Keep each other on track
- Self-rescue
- Communicate better – no code silos
- Learn from each other – knowledge transfer
- Courage to tackle tough problems
- There is research…



# TEST DRIVEN DEVELOPMENT

- We write tests before we write any code
- We only write the code we need so our tests pass
- Well tested code has fewer bugs
- Tests give us confidence to make changes

# TEST DRIVEN DEVELOPMENT

- Red – Write a failing test
- Green – Make it pass
- Refactor – Make it pretty

# RE-FRAMING FAILURE

- A failing test is the road to a passing test
- A failing test is our safety net
- A failing test gives us assurance
- A failing test is information

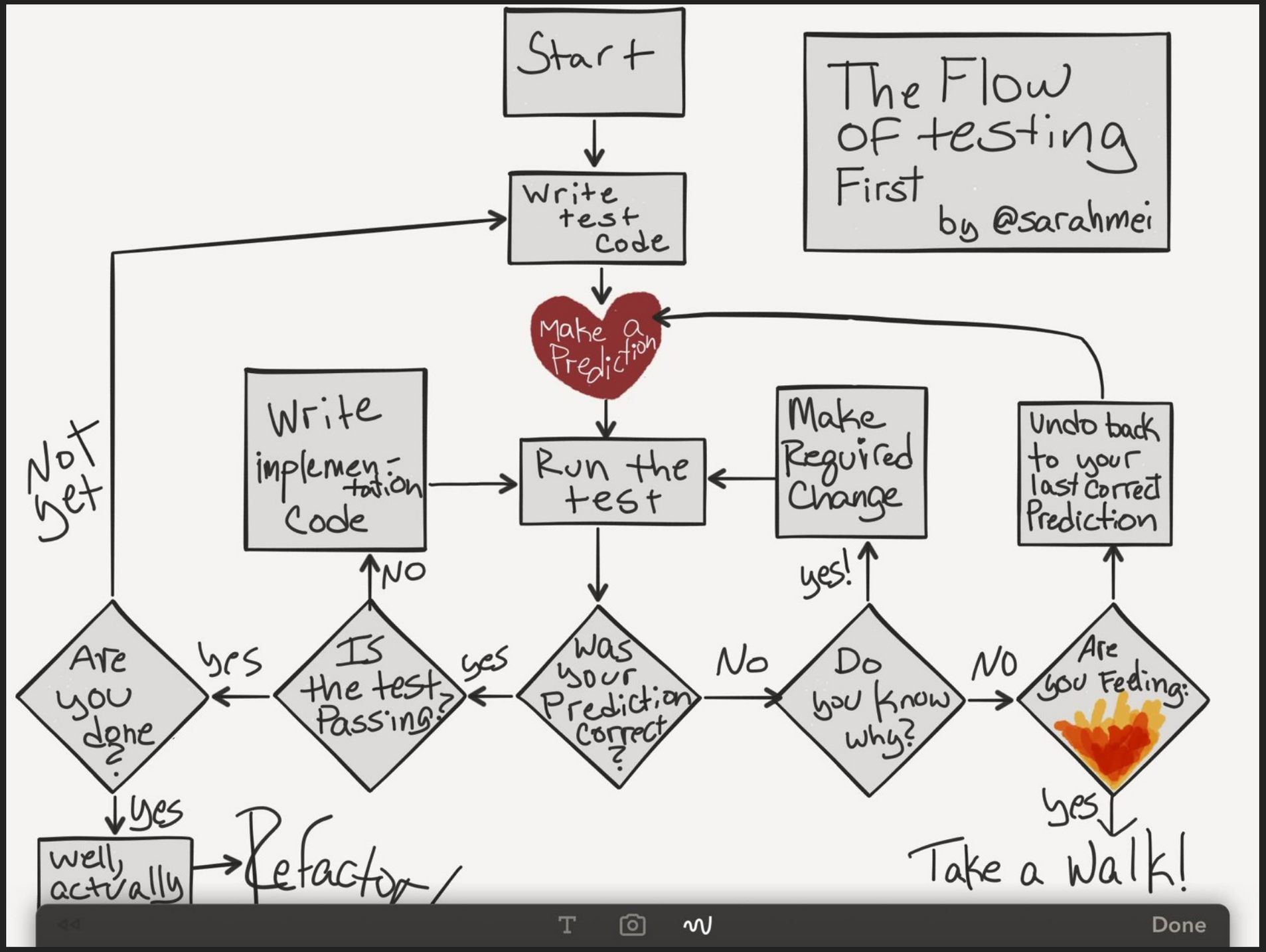
# TEST DRIVEN DEVELOPMENT

- Write a test
- Fail compile
- Write stub
- Fail test
- Write implementation
- Pass test
- Refactor
- Pass test
- Commit

# DO THE SIMPLEST THING FIRST

- It is ok to copy and paste
- It is ok to hard code a return value
- It is ok to write something ugly
- This is why we refactor

# TEST DRIVEN DEVELOPMENT



# DEVELOPER ENVIRONMENT

- JetBrains Editors (Webstorm for Javascript, Intellij Idea for java)
- Cloud Foundry CLI (CF Push, CI
- Git for VCS
- Gitx for visual diffs of commits
- Bluemix DevOps for Build & Deploy pipeline, DB services (DashDB or Compose PostGres)

# DEVOPS BUILD & DEPLOY

- Environment variables can be used for private keys or passwords that can’t be stored in a repo
- Jenkins under the hood for CI

# LET'S PAIR!

- Find a pair
- Each person turns on their laptop
- Get Sanity Tests Passing: http://jsfiddle.net/marlenac/wf75rmdy/
- Calculator Kata: http://jsfiddle.net/marlenac/xaqk5uop/
- Run a timer, each person types for 2 minutes

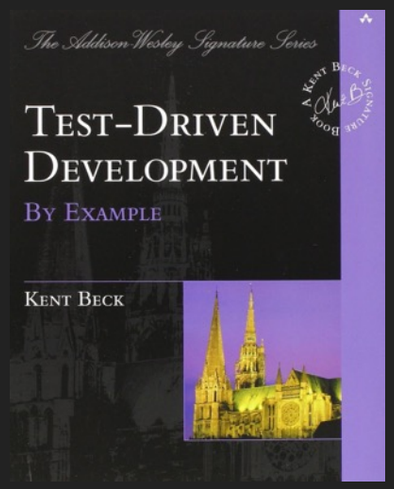
# TIPS

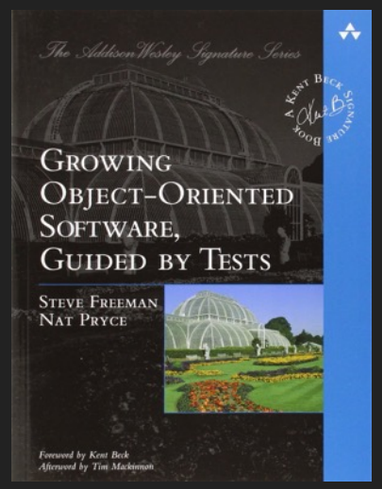
- Make a change…run the test…read the feedback
- Make a change…run the test…read the feedback
- If you get stuck, go back to your last predictable place when you knew exactly what was happening
- If you get a blank test-runner it’s a syntax problem with parenthesis or something.
- Use the inspector to play with javascript syntax
- If your pair is stuck, talk to another pair.

# GOING FURTHER WITH TDD

- Learning with TDD Katas http://www.peterprovost.org/blog/2012/05/02/kata-the-only-way-to-learn-tdd/
- Justin Searl's Test Wiki https:/ /github.com/testdouble/contributing-tests/wiki

- The two books:



# REFERENCES


- Agile Manifesto http://agilemanifesto.org/
- XP Explained http://www.amazon.com/Extreme-Programming-Explained-Embrace-Edition/dp/0321278658
- Pair Programming Illuminated http://www.amazon.com/Pair-Programming-Illuminated-Laurie-Williams/dp/0201745763/ref=sr_1_1?s=books&ie=UTF8&qid=1450471337&sr=1-1&keywords=pair+programming+illuminated
- Lean Startup http://www.amazon.com/Lean-Startup-Entrepreneurs-Continuous-Innovation/dp/0307887898/ref=sr_1_1?s=books&ie=UTF8&qid=1450471380&sr=1-1&keywords=lean+eric+ries
- User Story Mapping http://www.amazon.com/User-Story-Mapping-Discover-Product/dp/1491904909