Thinking and Discovering by Writing

Stephen Mizell

What is the first thing you do when you start a project?

Writing should be the first thing

Think about what you're going to do before doing it

Write about what you're going to do before doing it

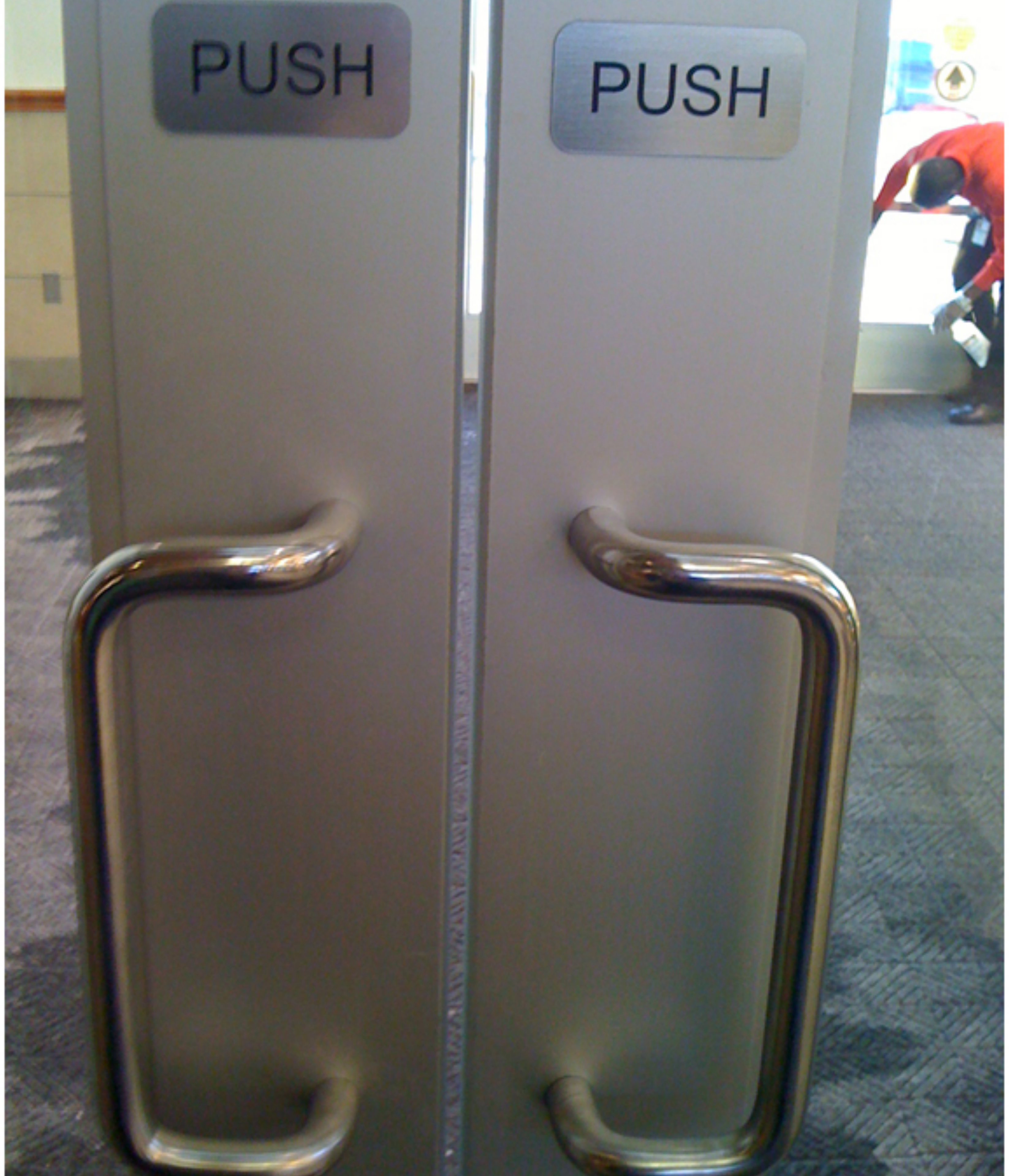
Writing Stages

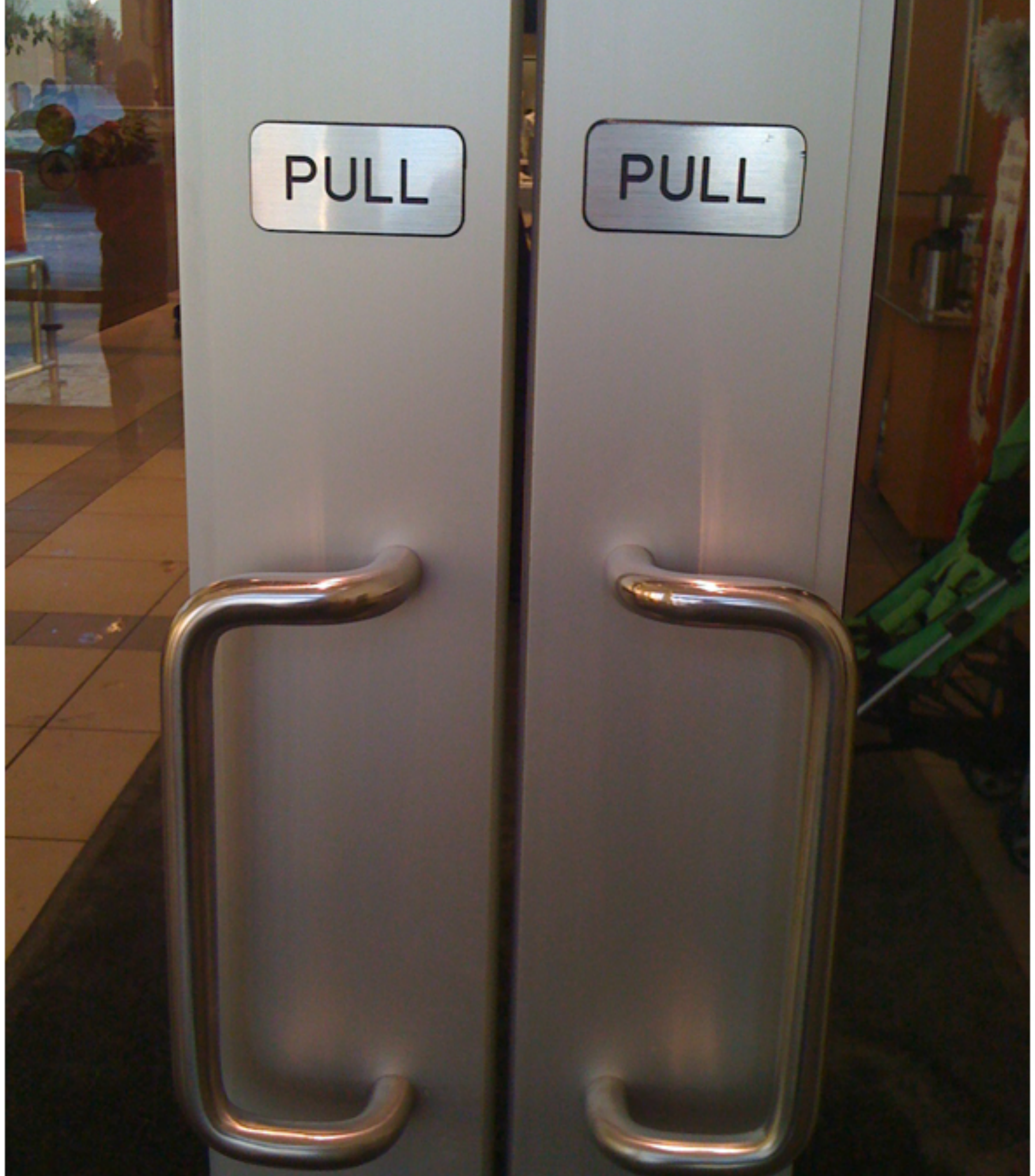
- Design
- User Stories
- Implementation
- Documentation

Design

Design the way the world works

- "Design of Everyday Things" by Donald Norman
- We know when something we encounter is wrong
- "Norman Doors"


PUSH
PUSH


PULL
PULL

Write the way the world works

Example: designing a door

What did we design?

- A physical door?
- A door application?
- A door API?

State Machine with Affordances

State Machine

- States and affordances for designers to use
- Foundation for user stories to drive implementation
- States and affordances to convey in API responses

Ubiquitous Language

Ubiquitous Language

By using the model-based language pervasively and not being satisfied until it flows, we approach a model that is complete and comprehensible, made up of simple elements that combine to express complex ideas.

— Eric Evans

Ubiquitous Language

Domain experts should object to terms or structures that are awkward or inadequate to convey domain understanding; developers should watch for ambiguity or inconsistency that will trip up design.

— Eric Evans

But naming is hard

> There are two hard things in computer science: cache invalidation, naming things, and off-by-one errors.
>
> — Jeff Atwood

Summary on Writing for Design

- Writing helps you think about design
- Writing helps you surface a common vocabulary
- Writing helps you understand the application's logic
- Writing helps you have a discussion apart from implementation

User Stories

User Stories

- Describes a user, what they want, and why
- Captures a deliverable feature
- Template for capturing simple requirements

User Stories

- Independent
- Negotiable
- Valuable
- Estimatable
- Small
- Testable

From Write a Great User Story

As a <role>, I want <goal>
so that <benefit>

Given... When... Then...

Gherkin

```
Feature: Lockable doors for users with keys
  As a door own
  I want to be able restrict access through the door
  So that I can secure my area

  Scenario: User with key encounters locked door
    Given I am a user with a key
      And I encounter a locked door
    When I try to unlock the door
    Then The door should be unlocked
```

Summary on Writing User Stories

- Capture small chunks of requirements
- Understand logic and value added before implementation
- Communicate with non-technical stakeholders

Implementation

Summary of Writing to Drive Implementation

- Understand logic and vocabulary before coding
- Defined scope of project requirements
- Defined behavior to drive development

Documentation

Good Documentation

- **Introduction** - General concept, feel, and understanding
- **Explanation** - How to actually use your project/software
- **Reference** - Reference material for drilling down
- **Troubleshooting** - True frequently asked questions

From How Great Documentation Drives Developer Adoption

Documentation Practices

- Readme Driven Development
- Contract First
- Documentation lives alongside code
- Documentation is generated from code
- Documentation with Continuous Integration Build

Documentation Tools

- GitHub Pages and Jekyll
- Literate Programming
- Read the Docs
- Sphinx

Summary of Writing Documentation

- Provides help for your users
- Byproduct of good product design and implementation
- Part of your deliverable to the end users

References

Writing

- Markdown - for all kinds of writing
- API Blueprint - for APIs
- Cucumber - for Gherkin
- How to Write Well-Formed Stories

References

Software Architecture

- Martin Fowler
- Uncle Bob
- Ruby Midwest 2011 - Keynote: Architecture the Lost Years - Robert Martin

References

Documentation

- Read the Docs
- Sphinx
- Jekyll
- How GitHub uses GitHub to document GitHub