

# Platformy programistyczne: .NET i Java

---

WYKŁAD 3: HELLO WORLD

# W poprzednim odcinku

---

- Git ciągle jest git
- .NET Framework – wprowadzenie
- .NET is all around



# Agenda

---

Część 1:

- Agile Manifesto
- Software Craftsmanship
- Keep Calm(s) and DevOps

Część 2:

- DevOps w praktyce

Część 3:

- [zakres nieznany i niezapowiedziany]





2001

# Agile Manifesto

---

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

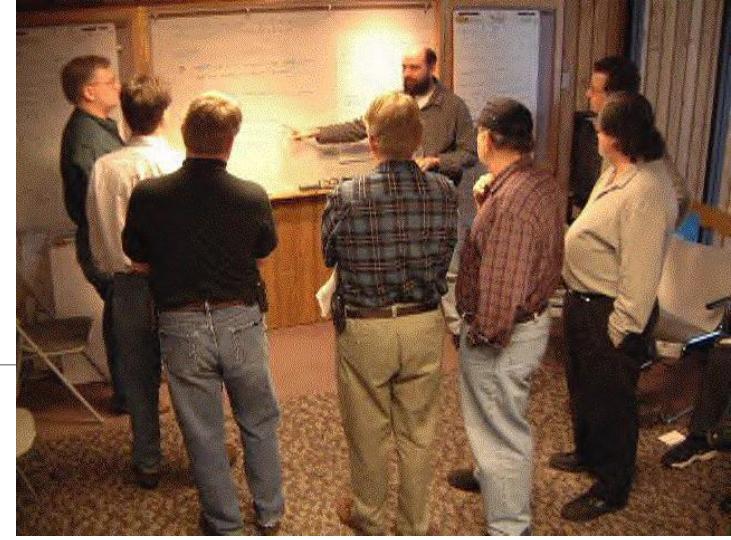
**Individuals and interactions** over processes and tools

**Working software** over comprehensive documentation

**Customer collaboration** over contract negotiation

**Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.



# Software Craftsmanship

---

As aspiring Software Craftsmen we are raising the bar of professional software development by practicing it and helping others learn the craft. Through this work we have come to value:

Not only working software, but also **well-crafted software**

Not only responding to change, but also **steadily adding value**

Not only individuals and interactions, but also **a community of professionals**

Not only customer collaboration, but also **productive partnerships**

That is, in pursuit of the items on the left we have found the items on the right to be indispensable.

```
<?php echo
```

```
"Hello World!"
```

```
?>
```



Do poczytania

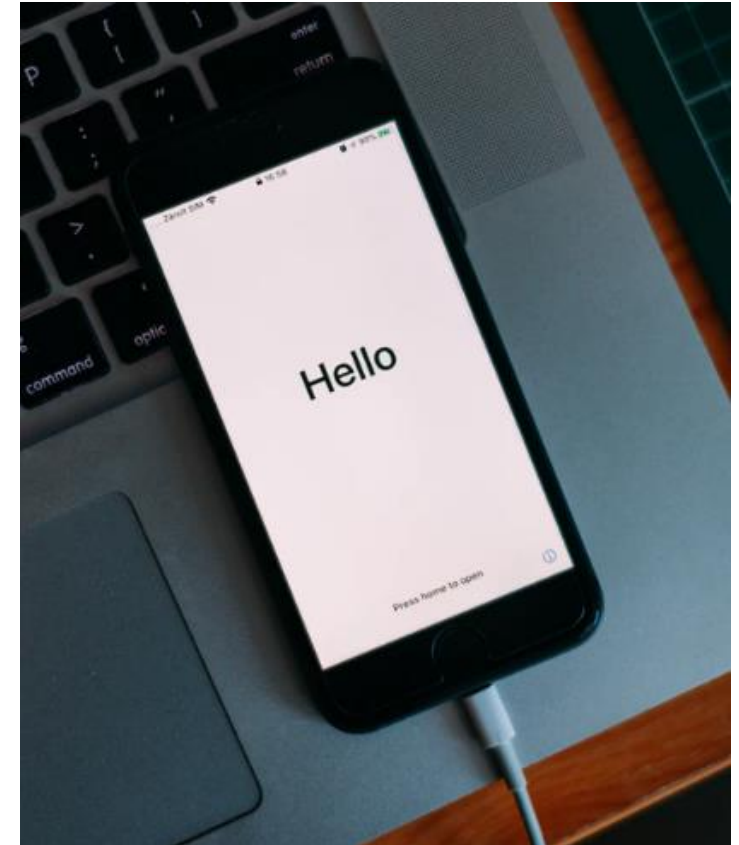
<https://stackoverflow.blog/2020/03/05/a-modern-hello-world-program-needs-more-than-just-code>

<http://helloworldcollection.de/>

# Hello world

---

- Separate project folder / Solution
- Source control
- Development tools
- Repeatable build process
- First line of code





# Keep it simple...

---



Gimme „Hello world” template...



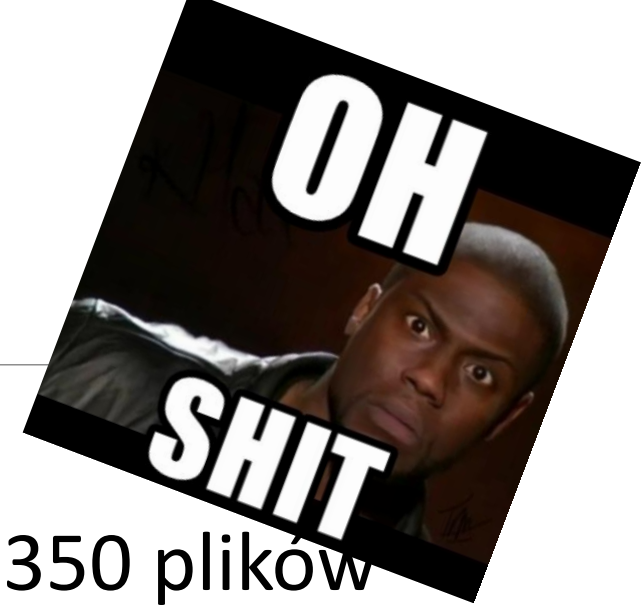
1 350 plików  
550 folderów



`create-react-app`



28 678 plików  
4 304 folderów



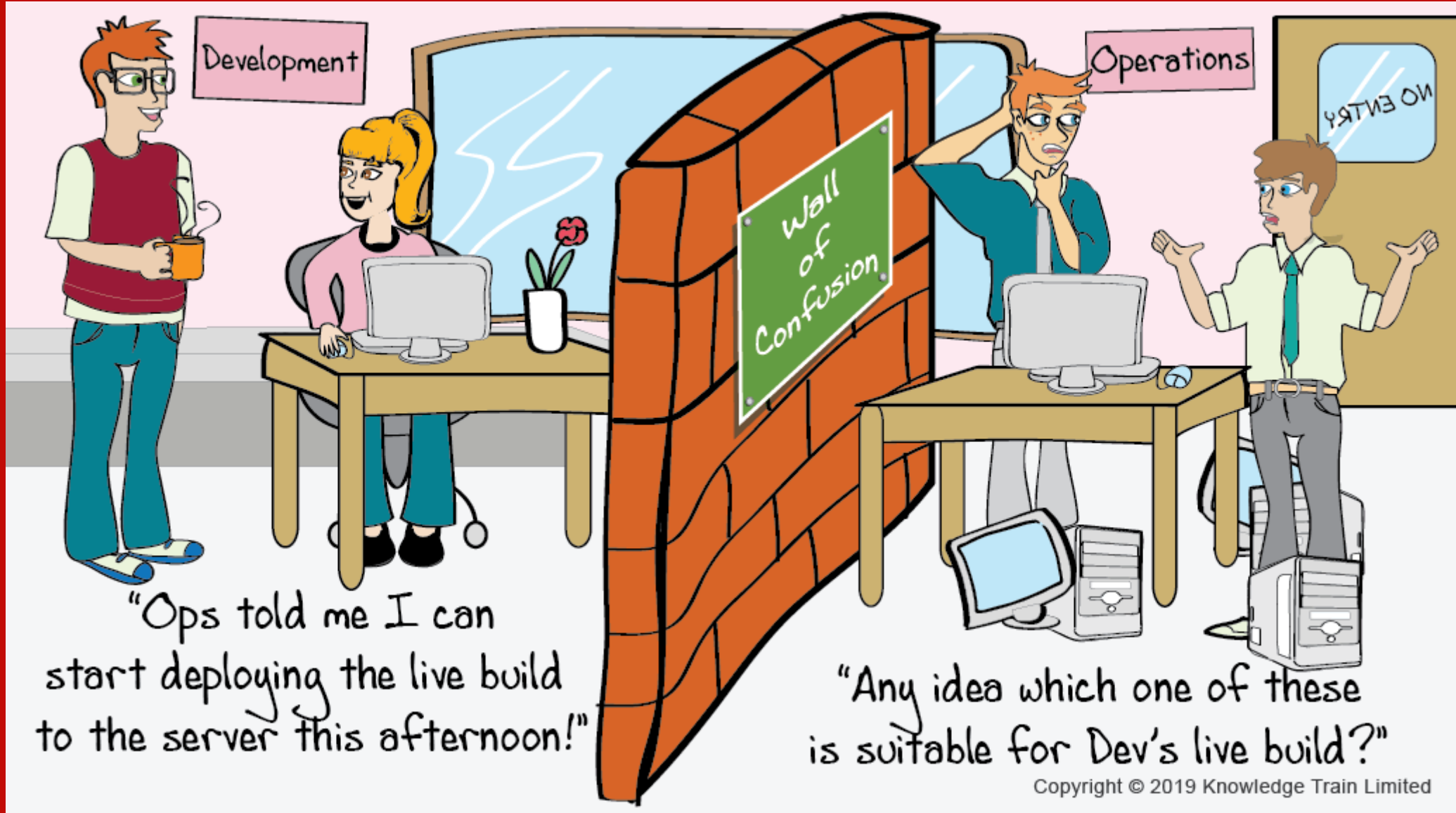


**KEEP  
CALMS  
AND  
DEVOPS**

**Culture**  
**Automation**  
**Lean**  
**Measurement**  
**Sharing**



# Culture



# Automation



# Lean



## TOO BUSY TO IMPROVE?

Haven't got time.  
We are busy delivering.

I've got an idea!



Chris Chan @ChrisChanAU <http://chrischan.com.au>  
Adapted from Hakan Forss @hakanforss



## The 8 Wastes

- Defects
- Overproduction
- Waiting
- Non-Utilized Talent
- Transportation
- Inventory
- Motion
- Extra Processing

# Measure



If you're having trouble already, lack of visibility won't help.

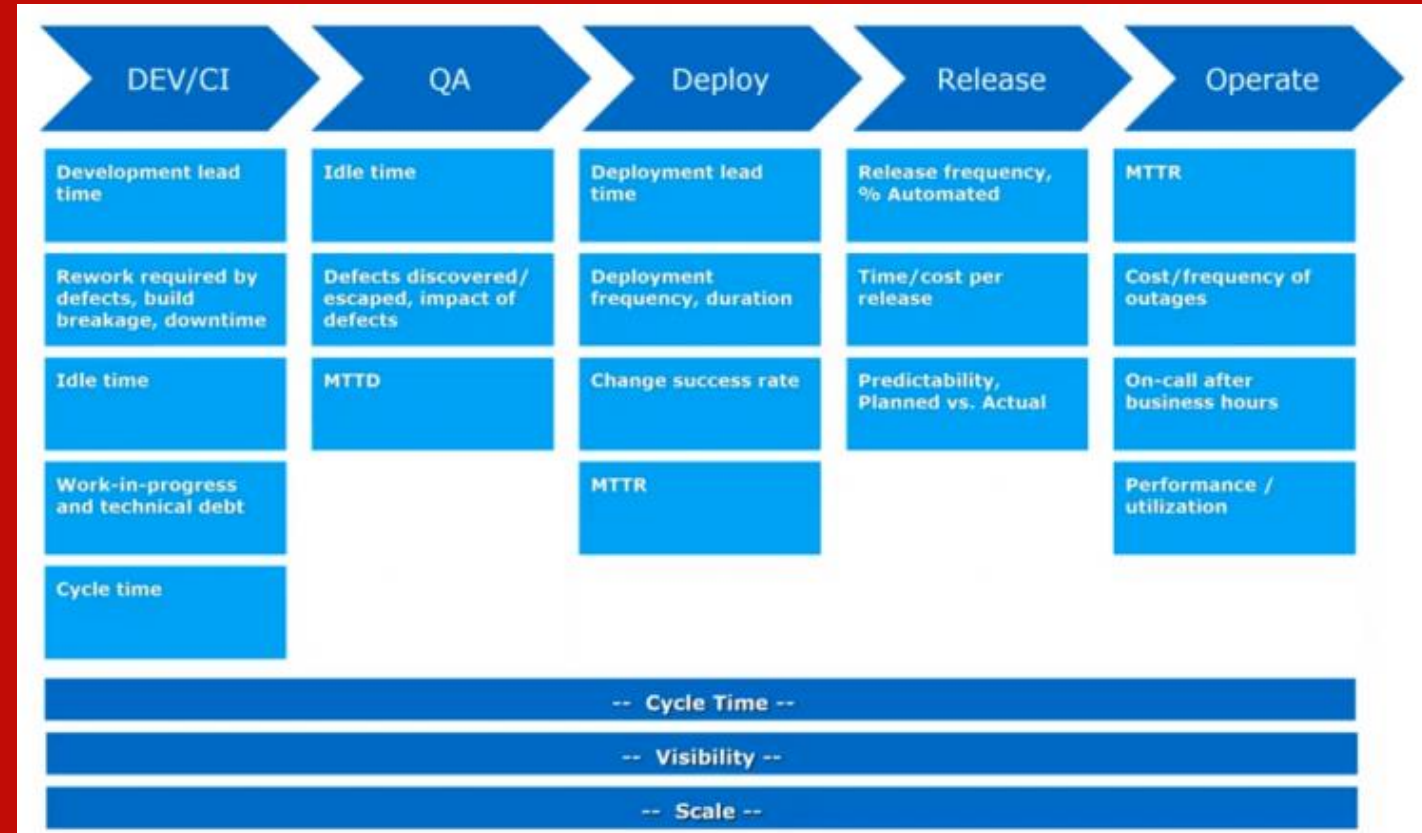
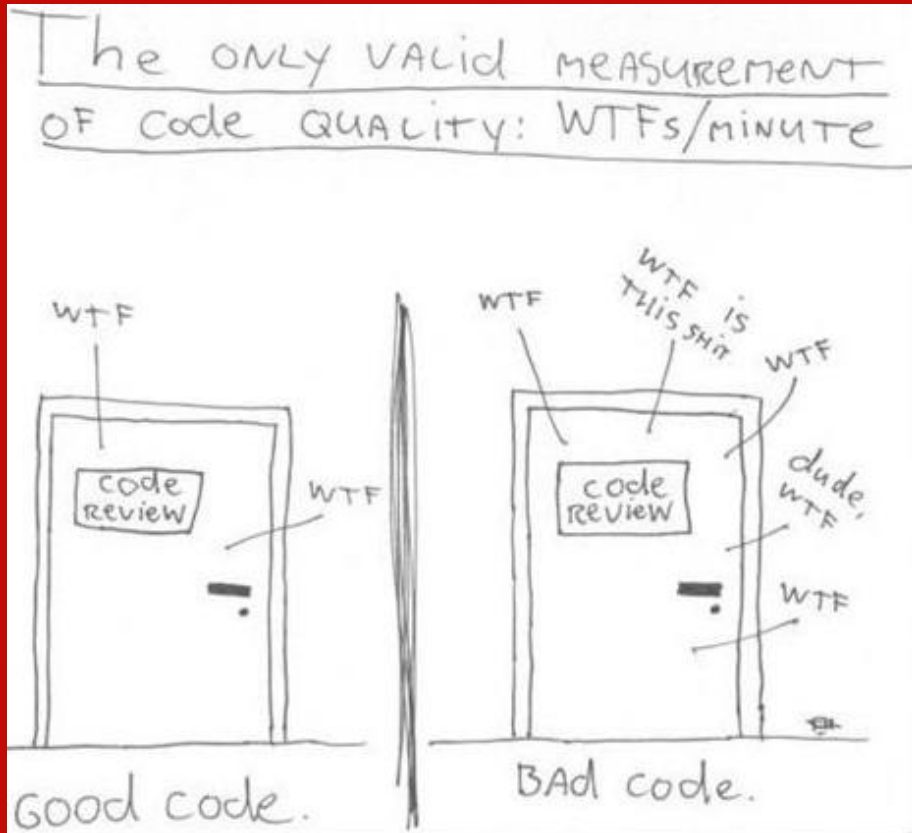
Anders Wallgren



# Measure

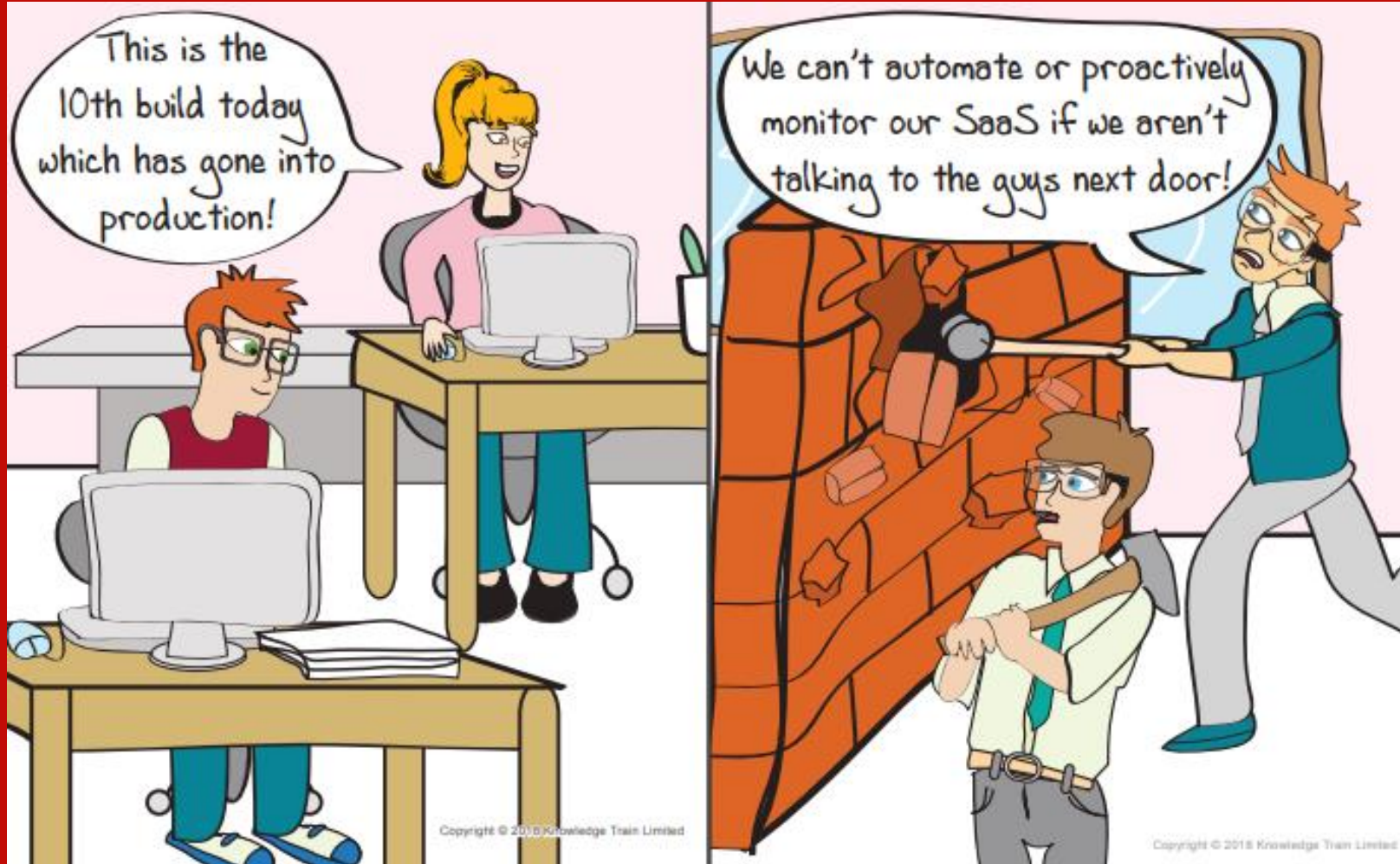


?





# Sharing



A Novel About IT,  
DevOps, and Helping  
Your Business Win

# The Phoenix Project

Gene Kim, Kevin Behr,  
and George Spafford



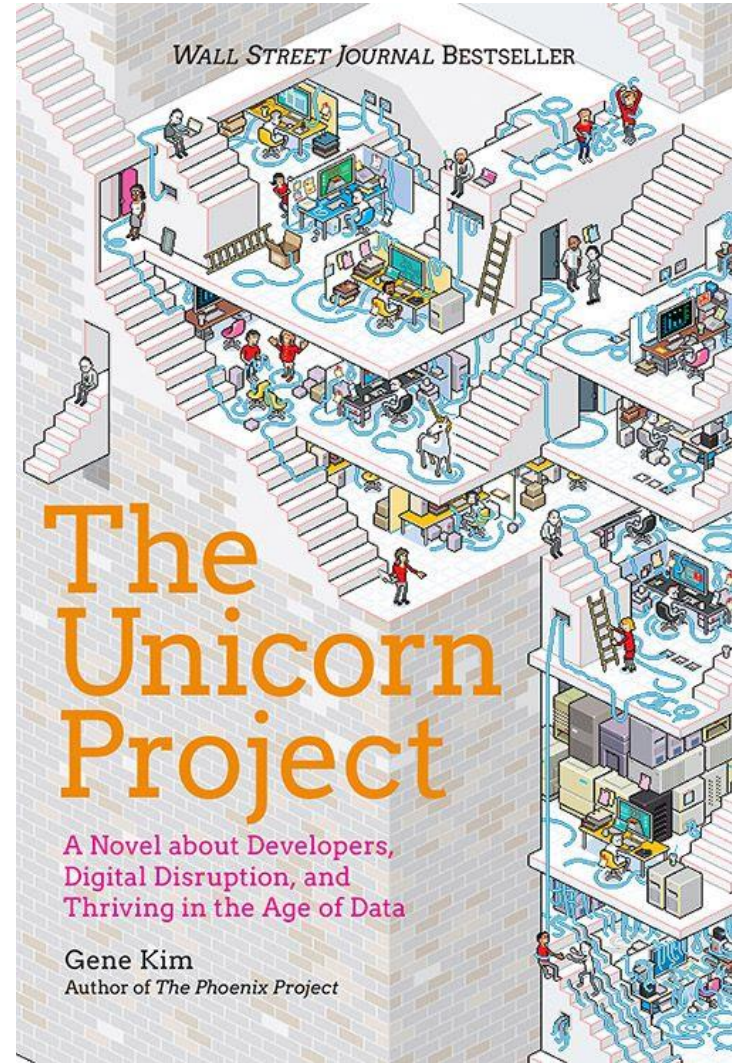
5<sup>TH</sup>  
ANNIVERSARY  
LIMITED  
EDITION

WALL STREET JOURNAL BESTSELLER

# The Unicorn Project

A Novel about Developers,  
Digital Disruption, and  
Thriving in the Age of Data

Gene Kim  
Author of *The Phoenix Project*



# The Five Ideals

---

THE FIRST IDEAL: Locality and Simplicity

THE SECOND IDEAL: Focus, Flow, and Joy

THE THIRD IDEAL: Improvement of Daily Work

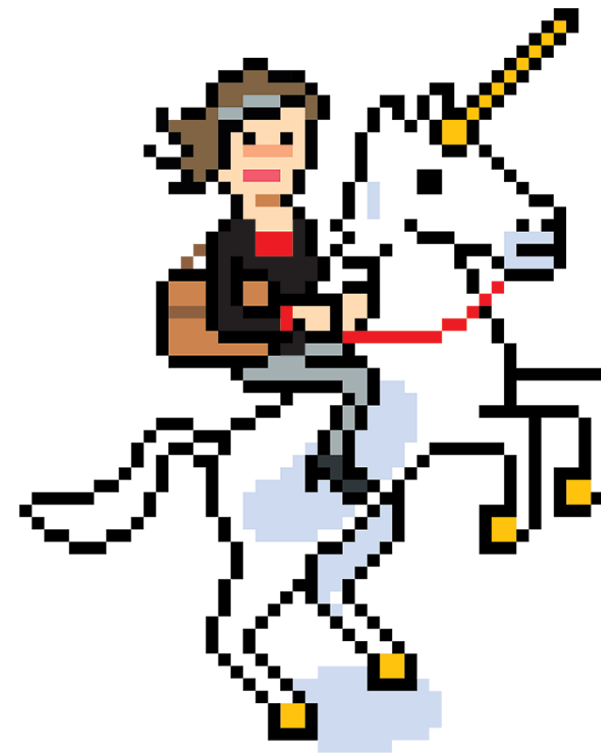
THE FOURTH IDEAL: Psychological Safety

THE FIFTH IDEAL: Customer Focus



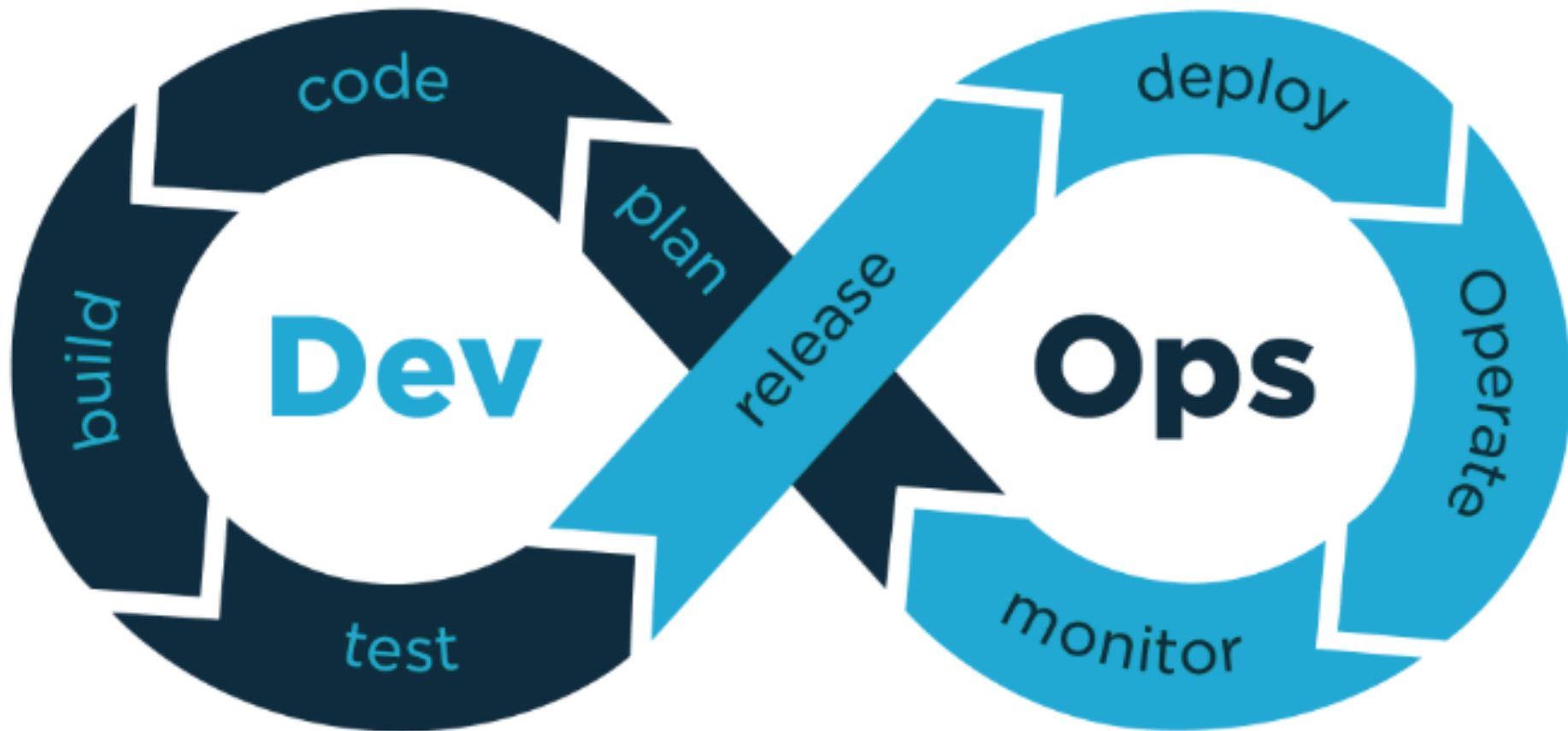
**Do pooglądania**

<https://www.youtube.com/watch?v=IOmoHCLDsG4&feature=youtu.be&t=2086>



# DevOps Cycle

---





## Version Control

Zarządzanie różnymi wersjami kodu



## Continuous Integration

Kompilacja, walidacja, zarządzanie Code Review, testy jednostkowe, testy integracyjne



## Continuous Delivery

Dostarczenie przygotowanej wersji do środowisk testowych. Wykonywanie testów akceptacyjnych



## Continuous Deployment

Dostarczenie przetestowanej aplikacji na środowiska produkcyjne.



### Do przeczytania

<https://dzone.com/articles/learn-how-to-setup-a-cicd-pipeline-from-scratch>



## Continuous Monitoring

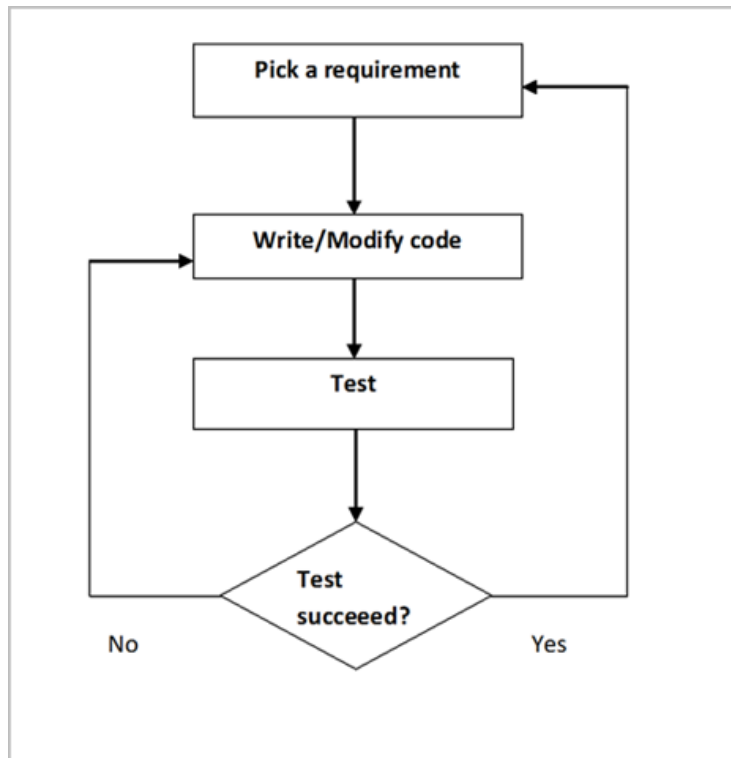


# Demo Time!

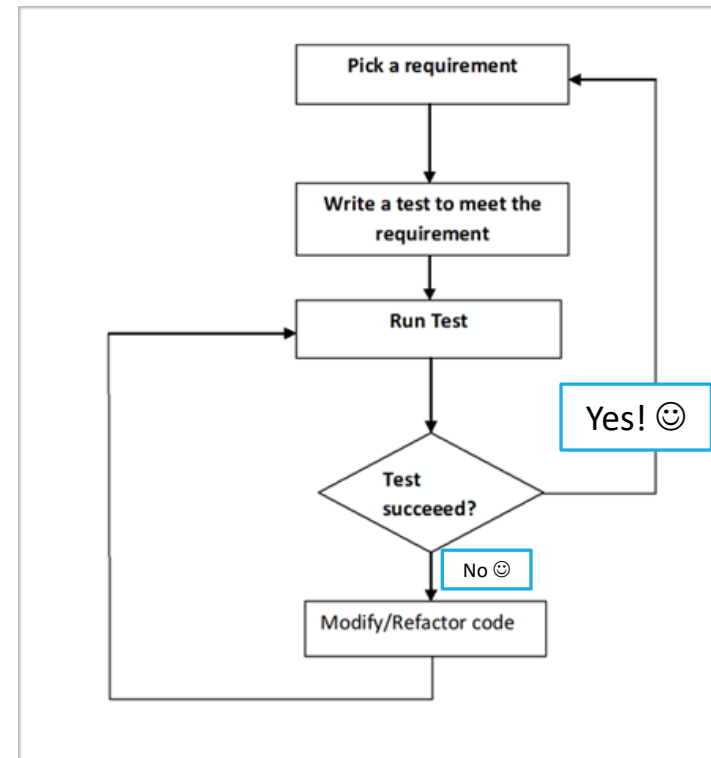
Azure DevOps time

# Test Driven Development

Tradycyjne podejście



Test Driver Development (TDD)



# Co daje podejście TDD?

---

- Testy automatyczne dla całego systemu
- Łatwiejsze tworzenie nowych funkcjonalności (stare pozostają stabilne i przetestowane)
- Większa pewność dla deweloperów (można spokojnie refaktoryzować)
- Skrócony czas testów manualnych
- Sposób dokumentacji systemu
- Sposób na fixowanie bugów
- Sposób na szybką detekcję powracających bugów w przyszłości



# Fizz Buzz Kata

---

- Przygotuj klasę / metodę, która wypisuje (zwraca) liczby
- Jeżeli liczba jest podzielna przez 3 – zwraca „Fizz”
- Jeżeli liczba jest podzielna przez 5 – zwraca „Buzz”
- Jeżeli liczba jest podzielna przez 3 i przez 5 – zwraca „FizzBuzz”

Czas na...



niezapowiedzianą kartkówkę...