

# Continuous Integration w praktyce

Czyli sam Jenkins to za mało



# Kto ja?

Łukasz Rybka

Team Leader / Senior Software Developer w Solwit S.A.

Trener / szkoleniowiec w infoShare Academy

Wykładowca na Politechnice Gdańskiej

# Podstawowe ustalenia

- Nie jestem alfą ani omegą
- Pytania mile widziane, szczególnie w trakcie!
- Slajdy to tylko notatki, roadmapa

# O czym porozmawiamy?

- Czym jest a czym nie jest Continuous Integration
- Jenkins CI
- Co poza serwerem CI nam potrzeba
- Wzorce i antywzorce związane z pracą w zespole
- Q&A

# Pytanie

Kto z Was pracuje zawodowo?

# Pytanie

Kto z Was używa na co dzień serwera Continuous Integration?

“ *A man who dares to waste one hour of time has not discovered the value of life*

*Charles Darwin*

“ *I choose a lazy person to do a hard job. Because a lazy person will find an easy way to do it.*

*Bill Gates*



# Początki



Kent Beck, 1999

Martin Fowler, 2006

- "Extreme Programming explained: Embrace Change", Kent Beck, 1999
- „Continuous Integration”, Martin Fowler, 2006

# Czy Continuous Integration to...

Narzędzie do kompilowania i linkowania kodu źródłowego?

# Czy Continuous Integration to...

Narzędzie do uruchamiania testów jednostkowych?

# Czy Continuous Integration to...

Narzędzie do wdrażania najnowszych zmian na środowiska developerskie i produkcyjne?

# Czy Continuous Integration to...

Narzędzie do przygotowywania binarnych wersji aplikacji?



# Czym jest Continuous Integration?

- Wywodzi się z Extreme Programming (XP)
- Praktyka w inżynierii oprogramowania
- Zakłada częste łączenie (integracje) kolejnych, mniejszych zmian w kodzie z branchem głównym
- Zakłada się kilka integracji dziennie, ~1 na jednego programistę w zespole

# Co nam daje Continuous Integration?

- Pozwala na uniknięcie tzw. „merging hell”
- Pozwala na szybkie wyłapanie błędów w architekturze
- Ułatwia rozmowę z Product Ownerem (klientem) i wspólną pracę nad funkcjonalnością
- Wprowadza nowy, wydajniejszy i bardziej ułożony sposób realizacji zadań



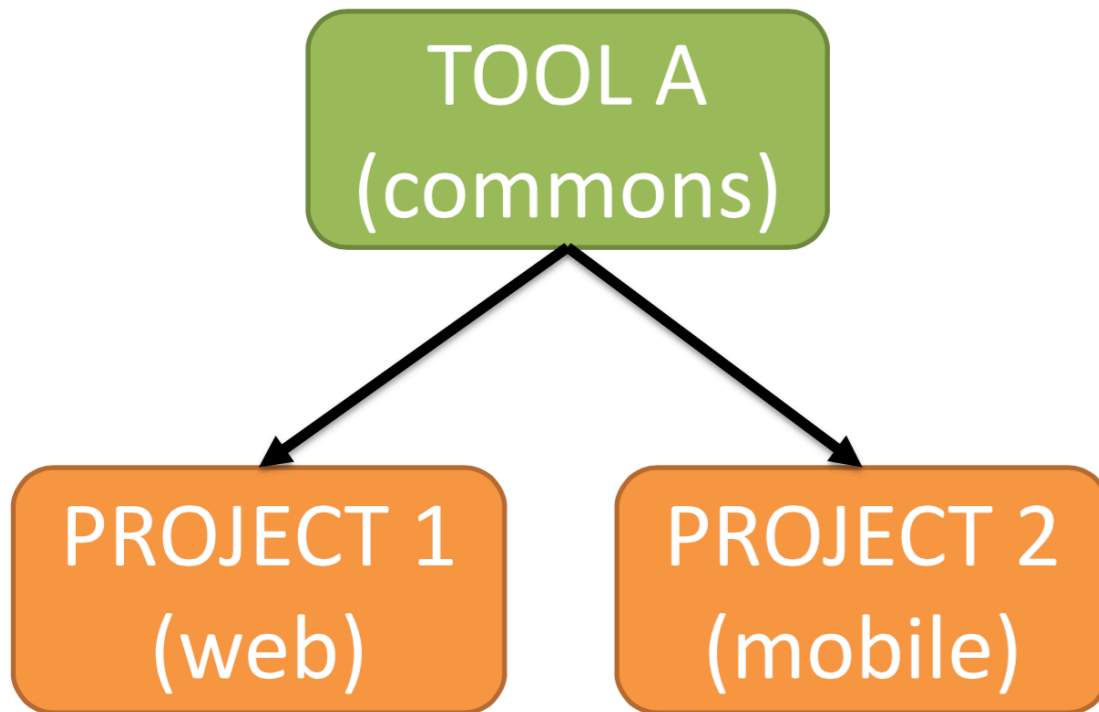
# Pytanie

Czy znacie Joela Spolsky'ego i jego test?

# Joel Test

1. **Do you use source control?**
2. **Can you make a build in one step?**
3. **Do you make daily builds?**
4. Do you have a bug database?
5. Do you fix bugs before writing new code?
6. Do you have an up-to-date schedule?
7. Do you have a spec?
8. Do programmers have quiet working conditions?
9. **Do you use the best tools money can buy?\***
10. Do you have testers?
11. Do new candidates write code during their interview?
12. Do you do hallway usability testing?

# Przykładowy proces budowania aplikacji



# Przykładowy proces budowania aplikacji

1. Kompilacja TOOL A
2. Uruchomienie testów jednostkowych TOOL A
3. Przygotowanie binarnej wersji TOOL A
4. „Przekopiowanie” binarnej wersji do PROJECT 1
5. Kompilacja PROJECT 1
6. Uruchomienie testów jednostkowych PROJECT 1
7. Wygenerowanie binarnej wersji PROJECT 1
8. „Przekopiowanie” binarnej wersji do PROJECT 2
9. Kompilacja PROJECT 2
10. Uruchomienie testów jednostkowych PROJECT 2
11. Wygenerowanie binarnej wersji PROJECT 2

# Przykładowy proces budowania aplikacji

1. Kompilacja TOOL A
  2. Uruchomienie testów jednostkowych TOOL A
  3. Przygotowanie binarnej wersji TOOL A
- FAZA 1
4. „Przekopiowanie” binarnej wersji do PROJECT 1
  5. Kompilacja PROJECT 1
  6. Uruchomienie testów jednostkowych PROJECT 1
  7. Wygenerowanie binarnej wersji PROJECT 1
- FAZA 2
8. „Przekopiowanie” binarnej wersji do PROJECT 2
  9. Kompilacja PROJECT 2
  10. Uruchomienie testów jednostkowych PROJECT 2
  11. Wygenerowanie binarnej wersji PROJECT 2
- FAZA 3

# Server Continuous Integration



Hudson



Jenkins

“ (...) I had this great idea of a web framework, and I wanted an excuse to actually implement it. This was around the time when everyone had his own framework, so I suppose I couldn't help myself.

*Kohsuke Kawaguchi*

# Jenkins - historia

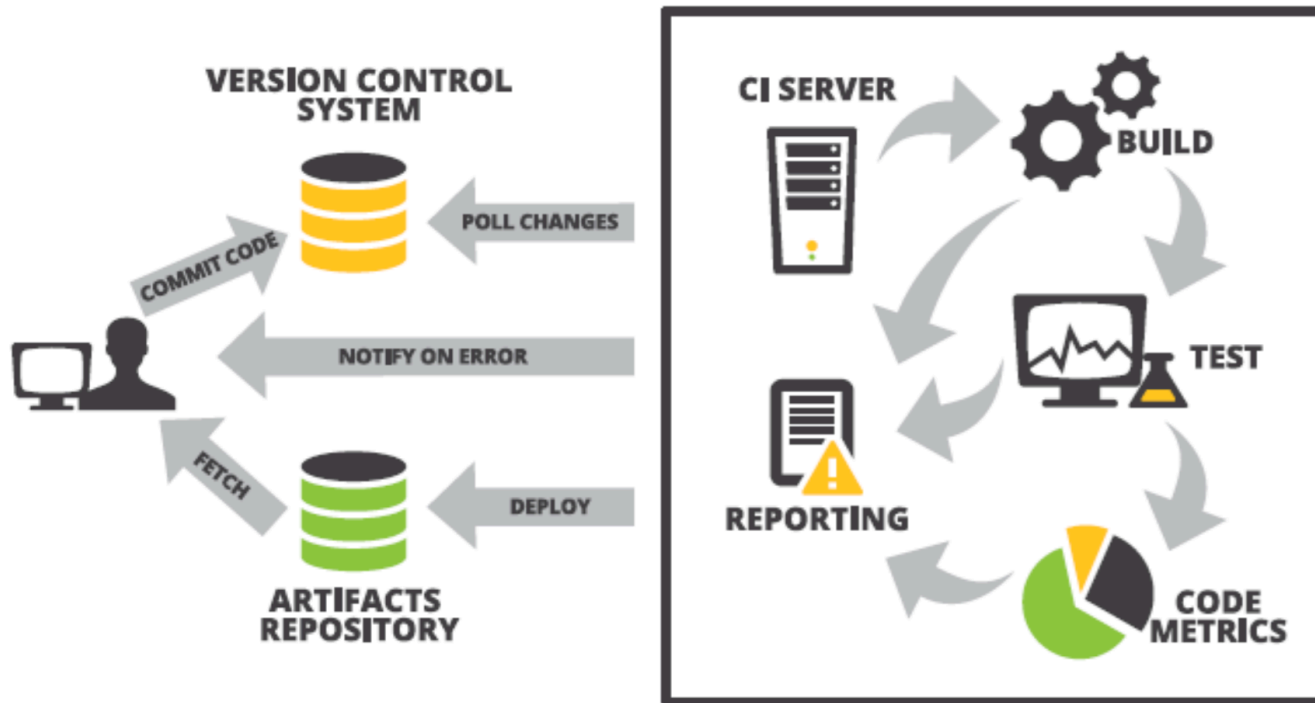
- 2004: rozpoczęcie prac przez Kawaguchi'ego nad projektem Hudson
- 2005: Hudson 1.0
- 2007: powstaje pierwszych 6 rozszerzeń do Hudsona
- 2008: ilość rozszerzeń rośnie do 27
- 2009: ilość rozszerzeń rośnie do 94
- 2010: Oracle przejmuje Sun Microsystems
- 2011: Kawaguchi odchodzi z Oracle, powstaje projekt Jenkins
- 2012: 469 rozszerzeń, ponad 30tys aktywnych instalacji Jenkinsa
- 2013: 636 rozszerzeń, ponad 50tys aktywnych instalacji



# Jenkins - podstawowe cechy

- Łatwa instalacja (plik war lub natywne paczki dla 9 systemów operacyjnych, w tym Windows, Linux oraz Max OS X)
- Łatwe i częste aktualizacje (release'y częściej niż raz w tygodniu)
- Ogromna baza pluginów (tworzonych przez społeczność)
- Prostota obsługi
- Możliwość łączenia wielu serwerów w „sieć” (master/slave) i automatyczne rozkładanie obciążenia i odpowiedzialności
- Niezależny od platformy i języka programowania
- Łatwość skryptowania
- Elastyczność

# Przykładowy proces budowania aplikacji



# Jenkins - konfiguracja

**Source Code Management**

- None
- CVS
- CVS Projectset
- Git
- Multiple SCMs

**Git**

Repositories

Repository URL  X ?

Credentials  Ad\*

Advanced...

Add Repository

Branches to build

Branch Specifier (blank for 'any')  X ?

Add Branch

Repository browser  ?

Additional Behaviours

**Check out to a sub-directory** X ?

Local subdirectory for repo  ?

Add

# Jenkins - konfiguracja

**Build Triggers**

- Trigger builds remotely (e.g., from scripts)
- Build after other projects are built
- Build periodically
- Poll SCM

Schedule

Would last have run at Tuesday, June 6, 2017 3:48:47 PM CEST; would next run at Tuesday, June 6, 2017 3:53:47 PM CEST.

Ignore post-commit hooks

# Jenkins - konfiguracja

### Build

#### Execute shell

Command `echo "gwt.sdk=/usr/share/gwt-2.5.1" > [REDACTED]/build.properties`  
`echo "gwt.sdk=/usr/share/gwt-2.5.1" > [REDACTED] build.properties`

See [the list of available environment variables](#)

Advanced...

#### Invoke Ant

Targets

Advanced...

#### Invoke Ant

Targets

Advanced...

Add build step ▾

# Jenkins - konfiguracja

**Editable Email Notification**
x
?

Disable Extended Email Publisher  ?

Allows the user to disable the publisher, while maintaining the settings

Project Recipient List ?

Comma-separated list of email address that should receive notifications for this project.

Project Reply-To List ?

Comma-separated list of email address that should be in the Reply-To header for this project.

Content Type ?

Default Subject ?

Default Content ?

Attachments ?

Attach Build Log ?

Content Token Reference ?

Advanced Settings...

Add post-build action ▾

Default Content Type ▾

\$DEFAULT\_SUBJECT

\$DEFAULT\_CONTENT

Tests: \${TEST\_COUNTS,var="pass"}\${TEST\_COUNTS,var="total"}

\${CHANGES}

Attach Build Log ▾

# Jenkins - konfiguracja

**Post-build Actions**

Trigger parameterized build on other projects

Build Triggers

Projects to build: [redacted] tests

Trigger when build is: Stable

Trigger build without parameters:

Add Parameters

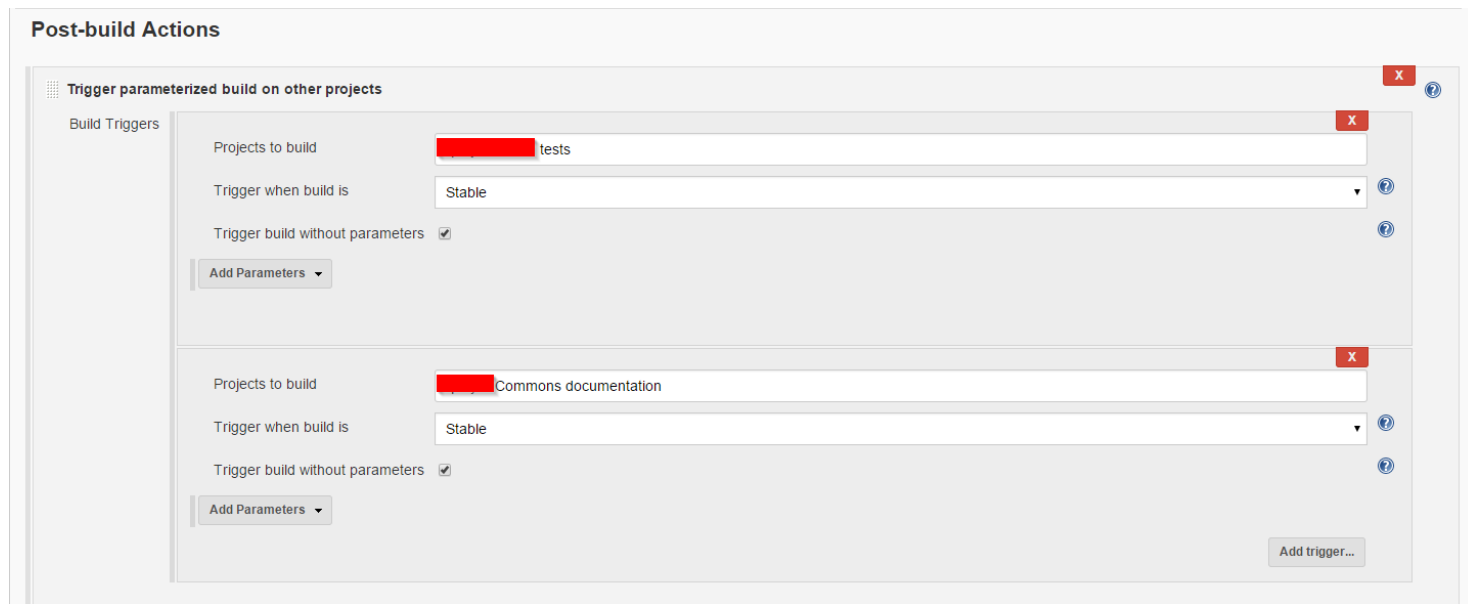
Projects to build: [redacted] Commons documentation

Trigger when build is: Stable

Trigger build without parameters:

Add Parameters

Add trigger...

The image shows a screenshot of the Jenkins 'Post-build Actions' configuration page. The main section is titled 'Trigger parameterized build on other projects'. Underneath, there is a 'Build Triggers' section. This section contains two identical configuration blocks for triggering other projects. Each block has a 'Projects to build' text input field (with a redacted name and 'tests' or 'Commons documentation'), a 'Trigger when build is' dropdown menu set to 'Stable', and a checked checkbox for 'Trigger build without parameters'. Below each block is an 'Add Parameters' button. At the bottom right of the configuration area, there is an 'Add trigger...' button.

# Jenkins - konfiguracja

**Build**

Execute shell

Command

```
rm -rf doc
yuidoc [redacted]/Commons/src -o doc
rm -rf /mnt/pub/[redacted]
cp -ar doc /mnt/pub/[redacted]
```

See [the list of available environment variables](#)

Advanced...

Add build step ▾



# Jenkins - konfiguracja

This project is parameterized

**String Parameter** X ?

Name  ?

Default Value  ?

Description  ?

[Safe HTML] [Preview](#)

**String Parameter** X ?

Name  ?

Default Value  ?

Description  ?

[Safe HTML] [Preview](#)

**String Parameter** X ?

Name  ?



Default Value  ?

Description  ?






[Safe HTML] [Preview](#)



Add Parameter ▾

# Jenkins - historia

 **Build History** [trend](#) 

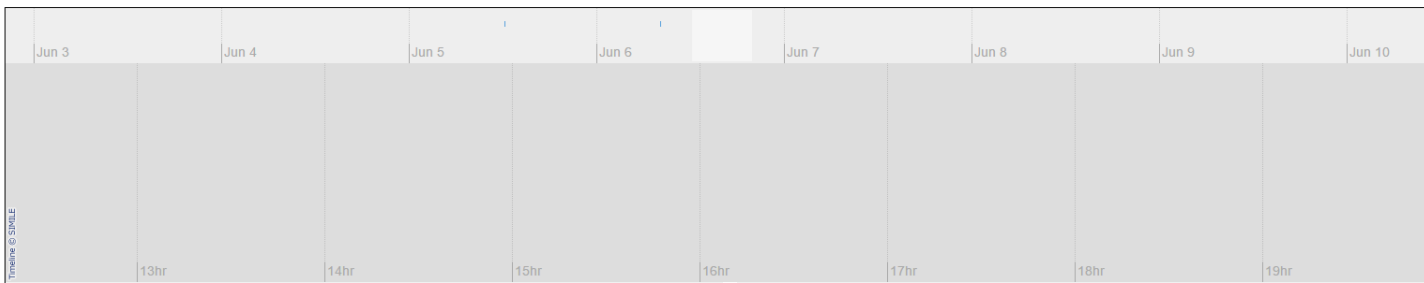
  

|   |                  |
|---|------------------|
|  <a href="#">#1277</a> | 2017-06-06 08:08 |
|  <a href="#">#1276</a> | 2017-06-05 12:19 |
|  <a href="#">#1275</a> | 2017-05-29 11:53 |
|  <a href="#">#1274</a> | 2017-05-26 16:13 |
|  <a href="#">#1273</a> | 2017-05-25 13:43 |

 [RSS for all](#)  [RSS for failures](#)

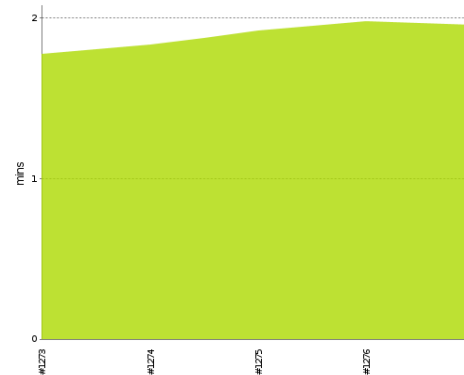
# Jenkins - historia

## Timeline



## Build Time Trend

| Build | Duration     |
|-------|--------------|
| #1277 | 1 min 57 sec |
| #1276 | 1 min 58 sec |
| #1275 | 1 min 55 sec |
| #1274 | 1 min 50 sec |
| #1273 | 1 min 46 sec |



# Czego poza serwerem CI potrzebujemy?

- Systemu kontroli wersji
- Narzędzia do budowania aplikacji
- Narzędzia do zarządzania zależnościami (wersjonowanymi)
- Narzędzia do przeprowadzania code review
- Języka skryptowego
- Repozytorium plików binarnych
- Platformy/środowiska testowego

# ”Sytuacja z życia wzięta” #1

„Proces jest nam niepotrzebny, mamy dwa tygodnie do deadline’u i musimy się na nim skupić” czyli o meksykańskich hotfix’ach

# „Sytuacja z życia wzięta” #2

„Moja zmiana jest tylko w pliku HTML, nie potrzebujemy robić tutaj review”  
czyli o tym jak po 3 minutach od wdrożenia rozdzwoniły się telefony.

# „Sytuacja z życia wzięta” #3

„Wprowadzenie środowiska testowego - stage - zredukowało czas potrzebny na wykonanie code/functional review o około 50%!”

# ”Sytuacja z życia wzięta” #4

„Dzięki automatyzacji procesu deploymentu czas, który developer musi poświęcić na niego został zredukowany o 85%!!!”



# ”Sytuacja z życia wzięta” #5

„Samo code review nam wystarczy”, czyli o tym jak recenzent przepuścił kod, który nawet się nie kompiluje ;)

# „Sytuacja z życia wzięta” #6

„Stopniowe wprowadzanie zmian do aplikacji pozwoliło błyskawicznie wykryć błędy i nieścisłości w oryginalnym projekcie, dzięki czemu zostały zaoszczędzone tygodnie pracy zespołu.”

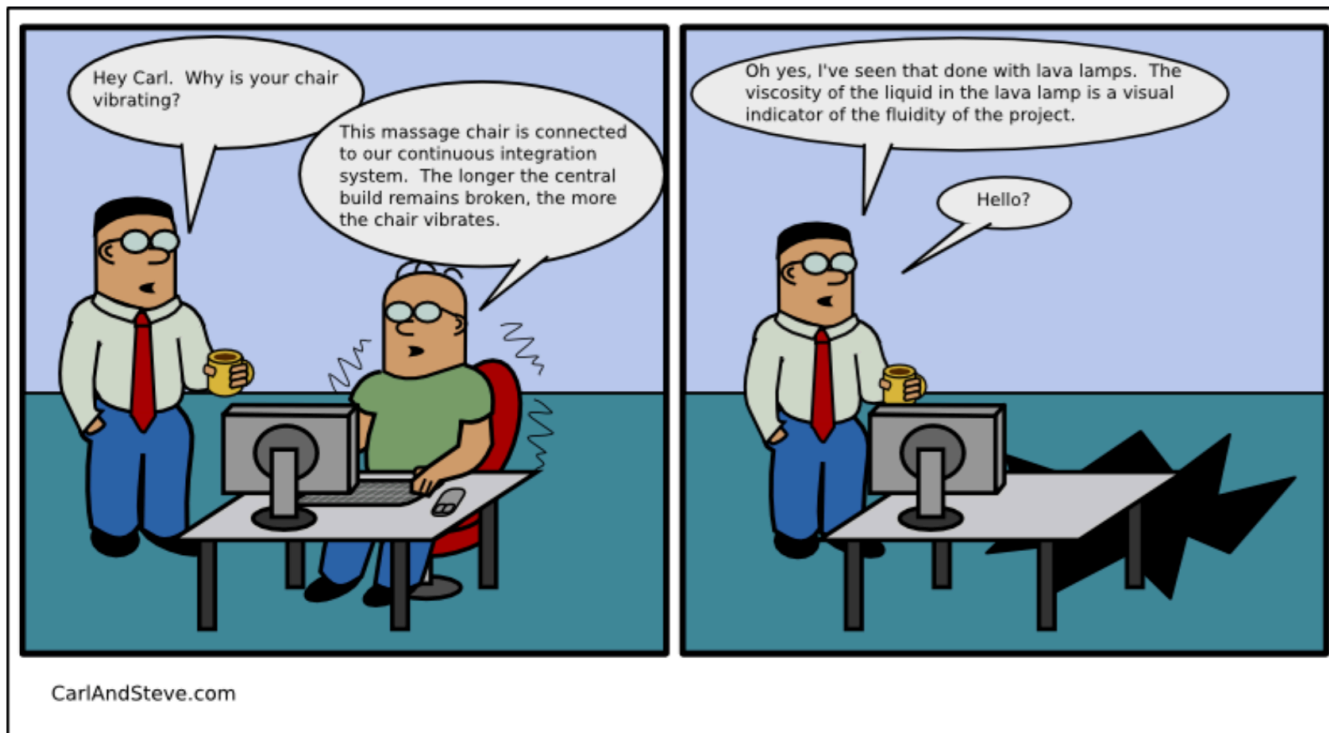
# ”Sytuacja z życia wzięta” #7

„Młody zrobił drop’a” czyli o tym jak nieświadomy nowy członek zespołu zrobił deployment niewłaściwej wersji aplikacji (złe środowisko).

# ”Sytuacja z życia wzięta” #8

Lokalny serwer CI, który pozwolił łatwo wykryć złamanie kompatybilności w wykorzystywanej bibliotece.

# Dygresja na koniec





# Materialy

- „The Joel Test: 12 Steps to Better Code”, Joel Spolsky, <http://www.joelonsoftware.com/articles/fog0000000043.html>
- „Daily Builds Are Your Friend”, Joel Spolsky, <http://www.joelonsoftware.com/articles/fog0000000043.html>
- „Why Devs <3 CI: A Guide to Loving Continuous Integration”, ZeroTurnaround, RebelLabs, <http://zeroturnaround.com/rebellabs/rebellabs-report-why-devs-love-ci-a-guide-to-loving-continuous-integration/>
- „Jenkins CI: The Origins of Butler, Build Masters and Bowties”, ZeroTurnaround, RebelLabs, <http://zeroturnaround.com/rebellabs/rebellabs-report-jenkins-ci-the-origins-of-butlers-build-masters-and-bowties/>

# Materialy

- „Feature Toggles”, Martin Fowler,  
<https://martinfowler.com/articles/feature-toggles.html>





# Thanks!