

# Эффективные методики автоматизированного тестирования в условиях непрерывной интеграции

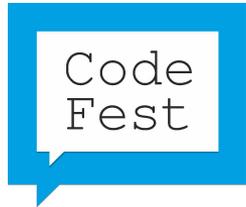


Сергей Андреев,  
JetBrains

Code  
Fest

# О себе (это очень важный слайд)

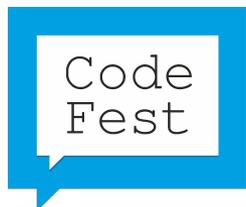




# Программы сами себя не напишут (с).

Проблемы:

- где рванёт после коммита?
- совместимы ли коммиты с жизнью приложения?
- тестерам постоянно нужна новая жертва
- как-то это все затягивается....



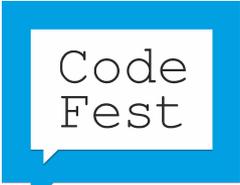
# CI - введение

*Непрерывная интеграция (Continuous Integration (CI)) – практика разработки ПО, когда все члены команды интегрируют результаты своей работы с некоторой частотой, обычно каждый участник интегрирует хотя бы раз в день, что приводит ко множественным интеграциям в течение дня.*

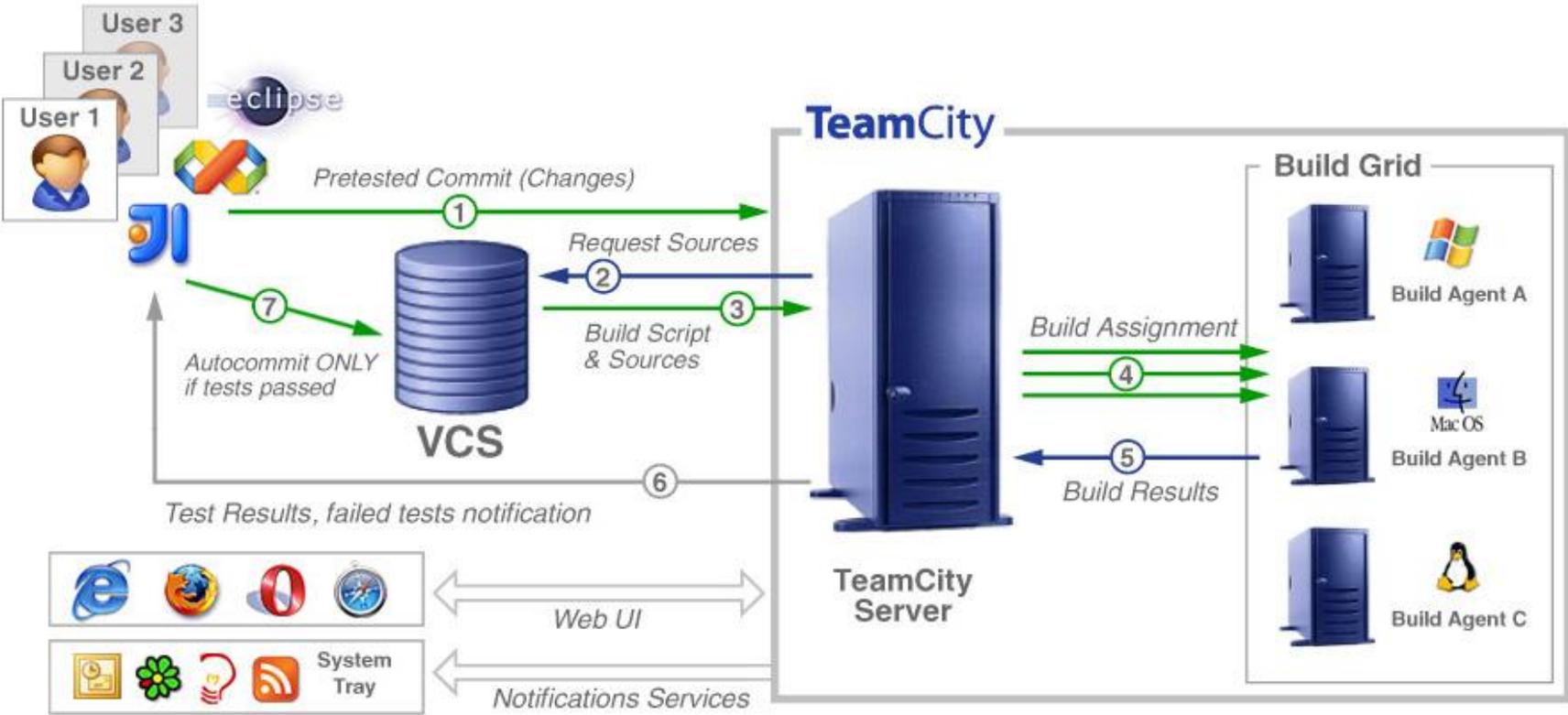
Вы хотите ссылок? Их есть у меня!

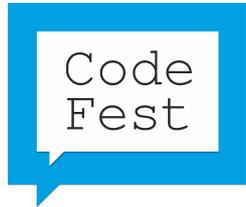
[http://en.wikipedia.org/wiki/Continuous\\_Integration](http://en.wikipedia.org/wiki/Continuous_Integration)

<http://www.martinfowler.com/articles/continuousIntegration.html>



# CI с иллюстрациями



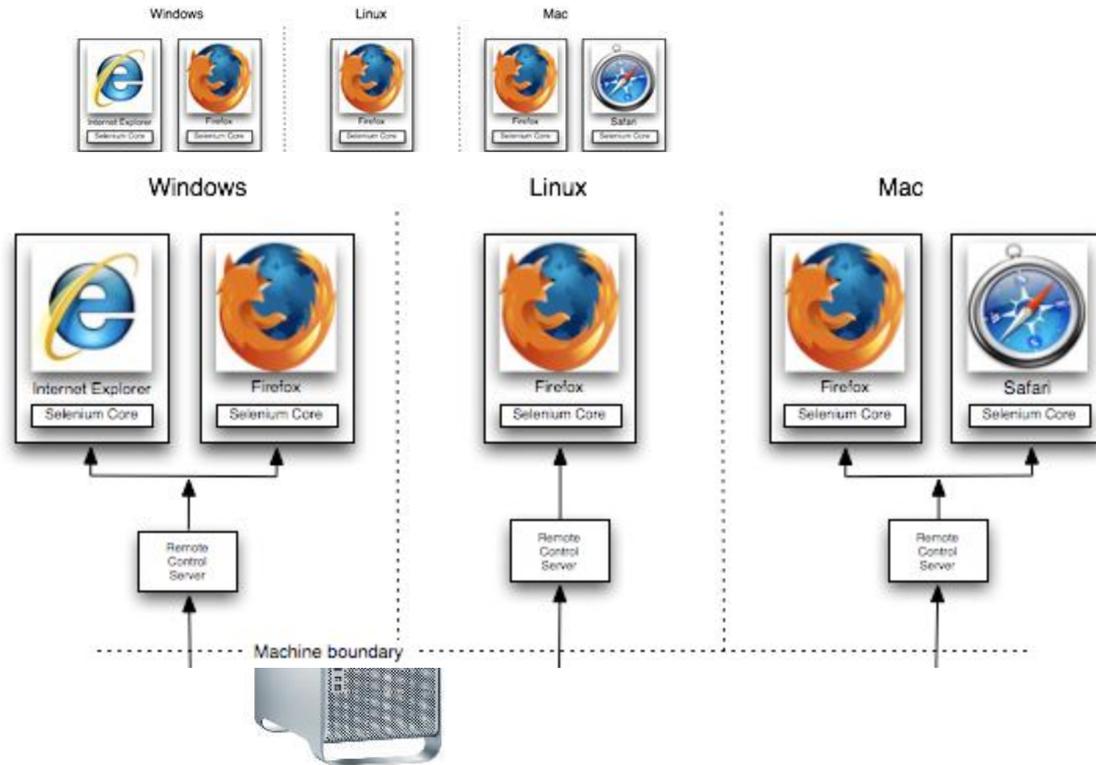


# В итоге:

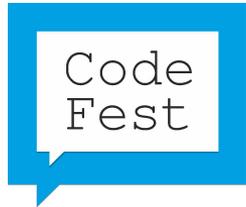
В итоге:

1. «Безопасность» изменений
2. Работоспособность приложения после изменений
3. Свежий билд для работы
4. ???
5. Экономия времени

# Как же без Selenium?



```
<jvmarg value="-Dhost=${seleniumhub.hostname}" />
<jvmarg value="-Durl=http://${env.HOSTNAME}:${web.port}" />
<jvmarg value="-Dbrowser=${browser}" />
```



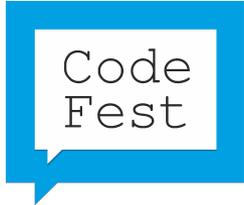
# Code Coverage

## COVERAGE SUMMARY FOR PACKAGE [jetbrains...

name	class, %	method, %	block, %	line, %
jetbrains...	60% (47/78)	56% (145/257)	40% (2681/6742)	45% (623.6/1371)

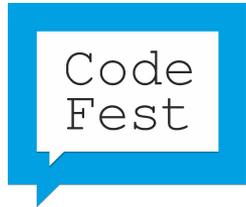
## COVERAGE BREAKDOWN BY SOURCE FILE

name	class, %	method, %	block, %	line, %
...	0%	0%	0%	0%
...	0%	0%	0%	0%
...	0%	0%	0%	0%
...	0%	0%	0%	0%
...	50%	20%	1%	8%
...	50%	20%	1%	8%
...	100%	50%	18%	21%
...	100%	67%	19%	24%
...	44%	33%	22%	24%
...	100%	67%	40%	43%
...	67%	59%	41%	38%
...	100%	100%	42%	46%
...	100%	100%	43%	45%
...	100%	100%	44%	60%
...	100%	79%	46%	47%
...	100%	50%	50%	50%
...	100%	100%	67%	64%
...	100%	50%	70%	33%
...	100%	89%	76%	60%
...	100%	100%	77%	82%
...	100%	100%	78%	80%
...	100%	67%	79%	67%
...	100%	86%	82%	76%
...	100%	67%	82%	88%
...	100%	67%	83%	80%
...	100%	100%	84%	83%
...	100%	80%	84%	86%



# Unit Tests





# JMeter

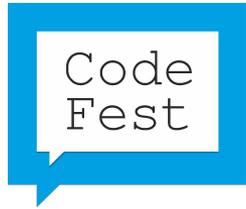
Вопрос залу: На сколько процентов JMeter лучше, чем ничего?

Ответ:

Как минимум в 2 раза больше.

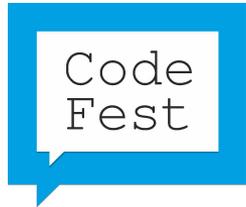
Функциональное тестирование RESTful интерфейса

- + Легко создавать тесты
- + Навыков программирования не нужно
- Тесты неуклюжие



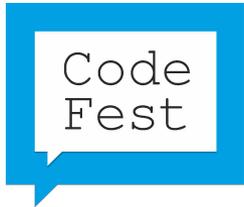
**Внезапно...**

**DSL**



# Ходят и пишут, что DSL...

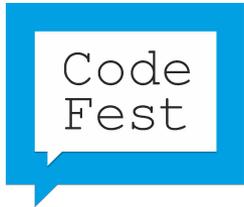
```
tst.openURL("http://www.google.com/webhp?complete=1&hl=en");  
  
tst.typeSmth("Cheese", "q");  
  
tst.waitForElementPresent("gac_m", 5000);
```



# DSL

**Предметно-ориентированный язык программирования** (англ. *domain-specific programming language, domain-specific language, DSL*) — язык программирования, специально разработанный для решения определённого круга задач, в отличие от языков программирования общего назначения, таких, как Си, или языков моделирования общего назначения наподобие UML, PostScript, SQL и др.

[http://ru.wikipedia.org/wiki/Предметно-ориентированный\\_язык\\_программирования](http://ru.wikipedia.org/wiki/Предметно-ориентированный_язык_программирования)



# DSL – JMeter Lang

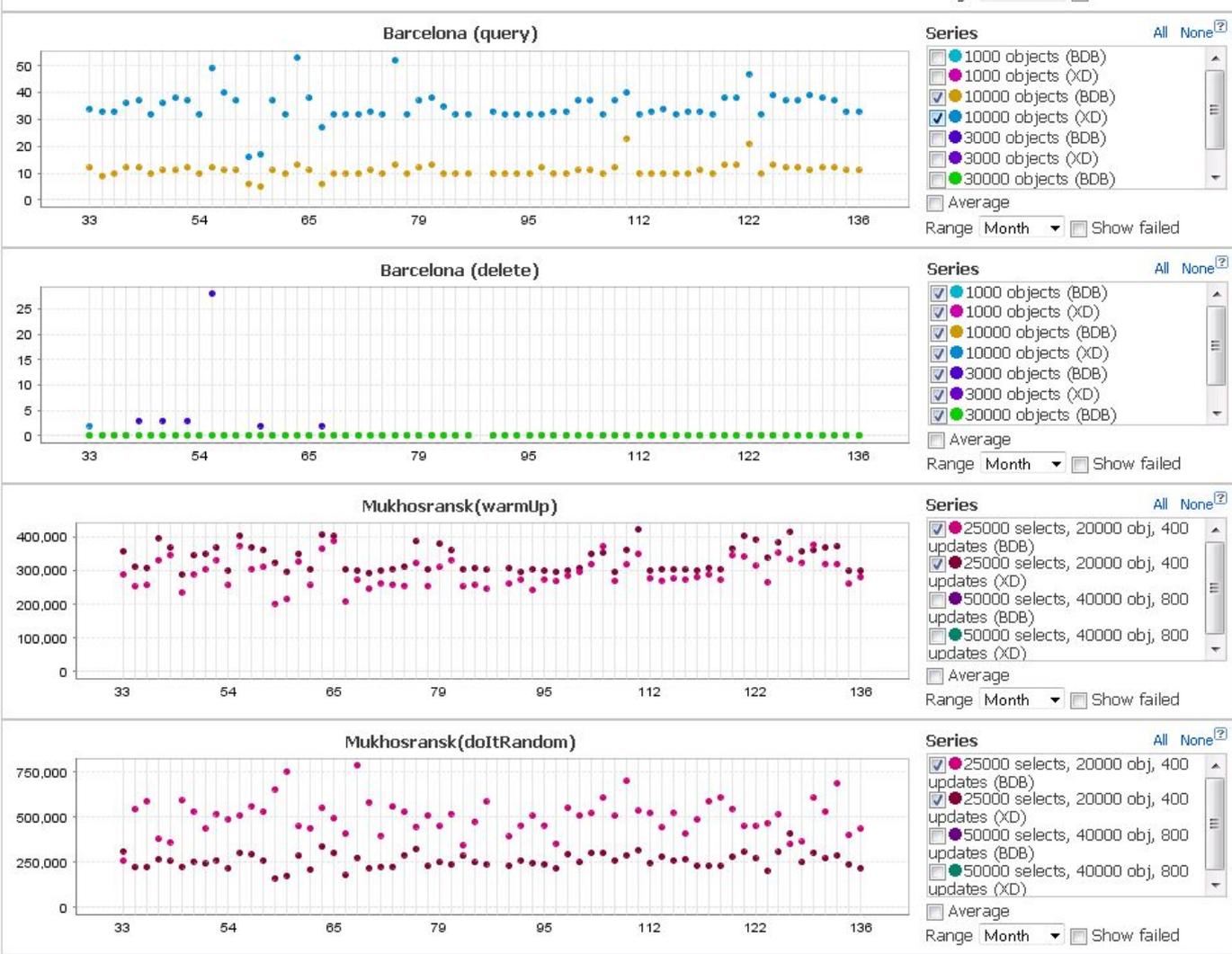
```
request Get all bundles for get {
  parameters:
    << no parameters >>
  assert:
    code 200: OK
    text equals <?xml version="1.0" encoding="UTF-8" standalone...
}

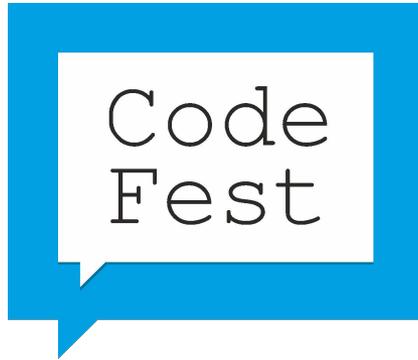
request Remove group from bundle for delete /{bundleName}/group/{fieldValue} {
  parameters:
    path bundleName = newBundleName
    path fieldValue = "idea-developers"
  assert:
    code 200: OK
}

request <no name> for <no method> {
  parameters:
    << no parameters >>
  assert:
    << no asserts >>
}
```



# PolePosition





Пожалуй хватит.  
Спасибо за внимание!



Сергей Андреев,  
JetBrains

[sergey.andreev@jetbrains.com](mailto:sergey.andreev@jetbrains.com)  
[smandreev@gmail.com](mailto:smandreev@gmail.com)