35.000 **tests** and counting!

Bill Kalligas
Web developer for e-Travel SA

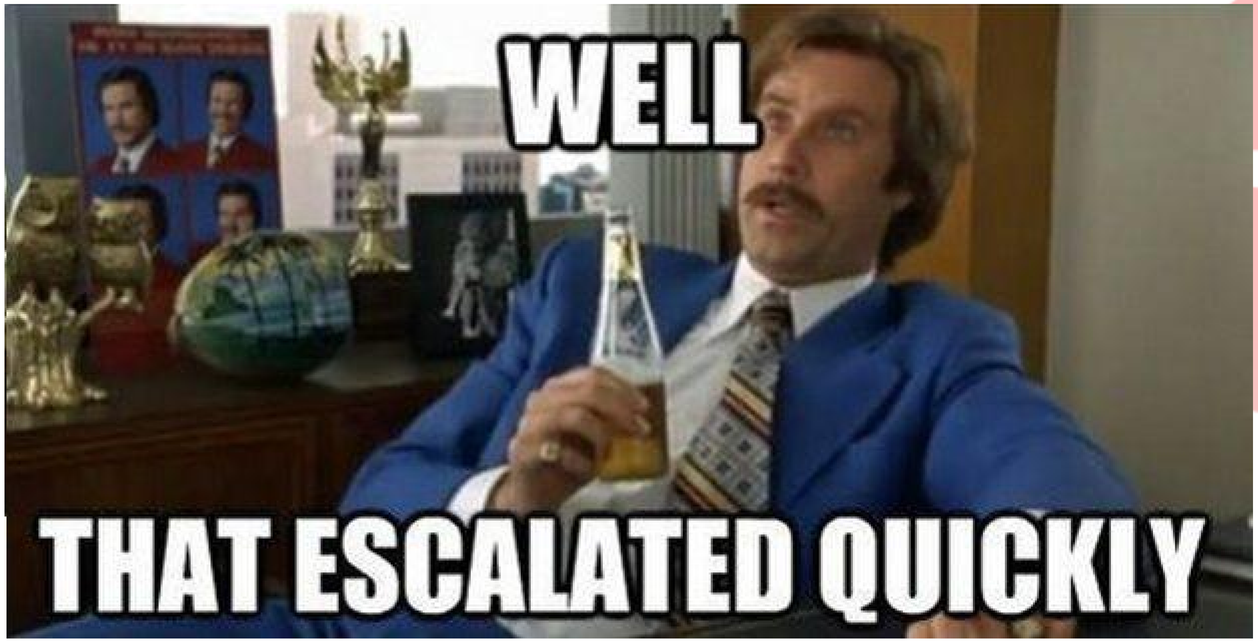
@billkall

Who we are

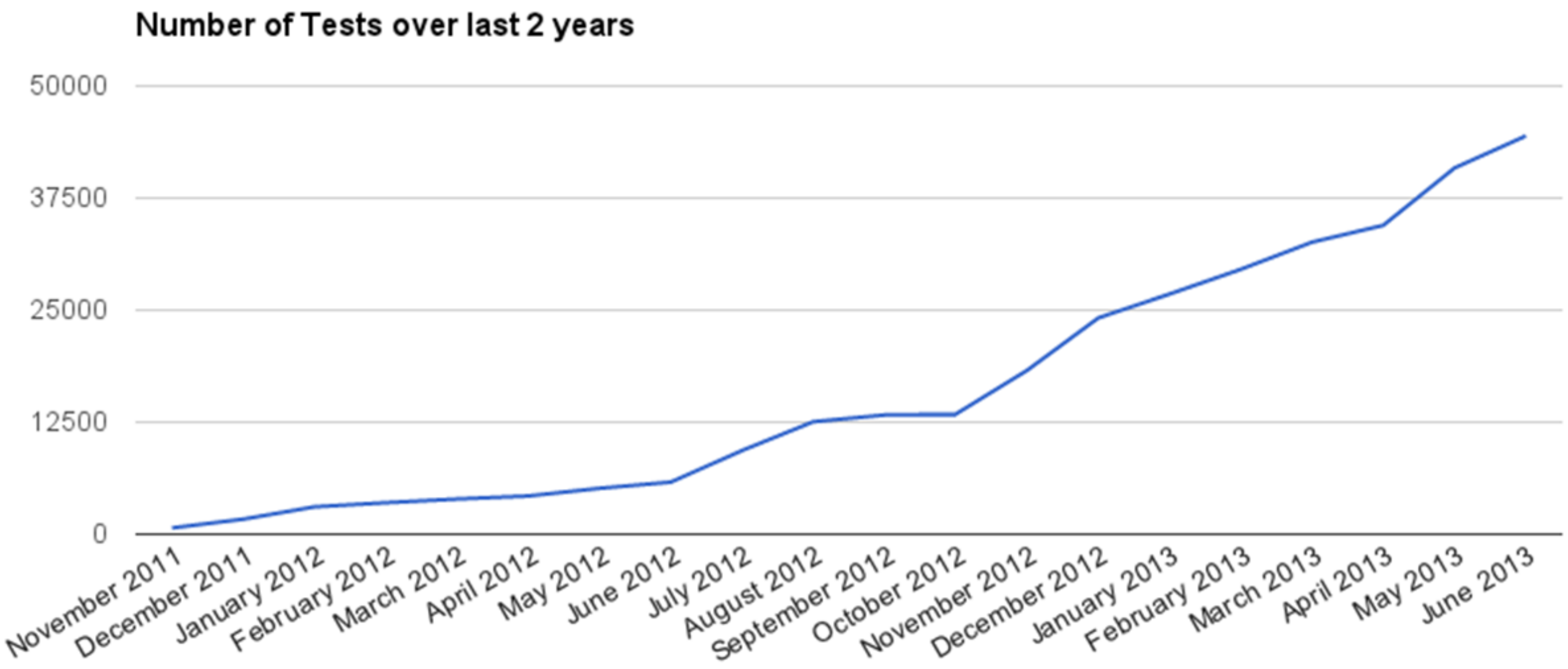
- e-Travel is one of the top 10 European Online Travel Agencies
- IT crowd of around 20 developers
- 4 of which are working on site's UI / UX
- 2 years' old project

Well… actually…

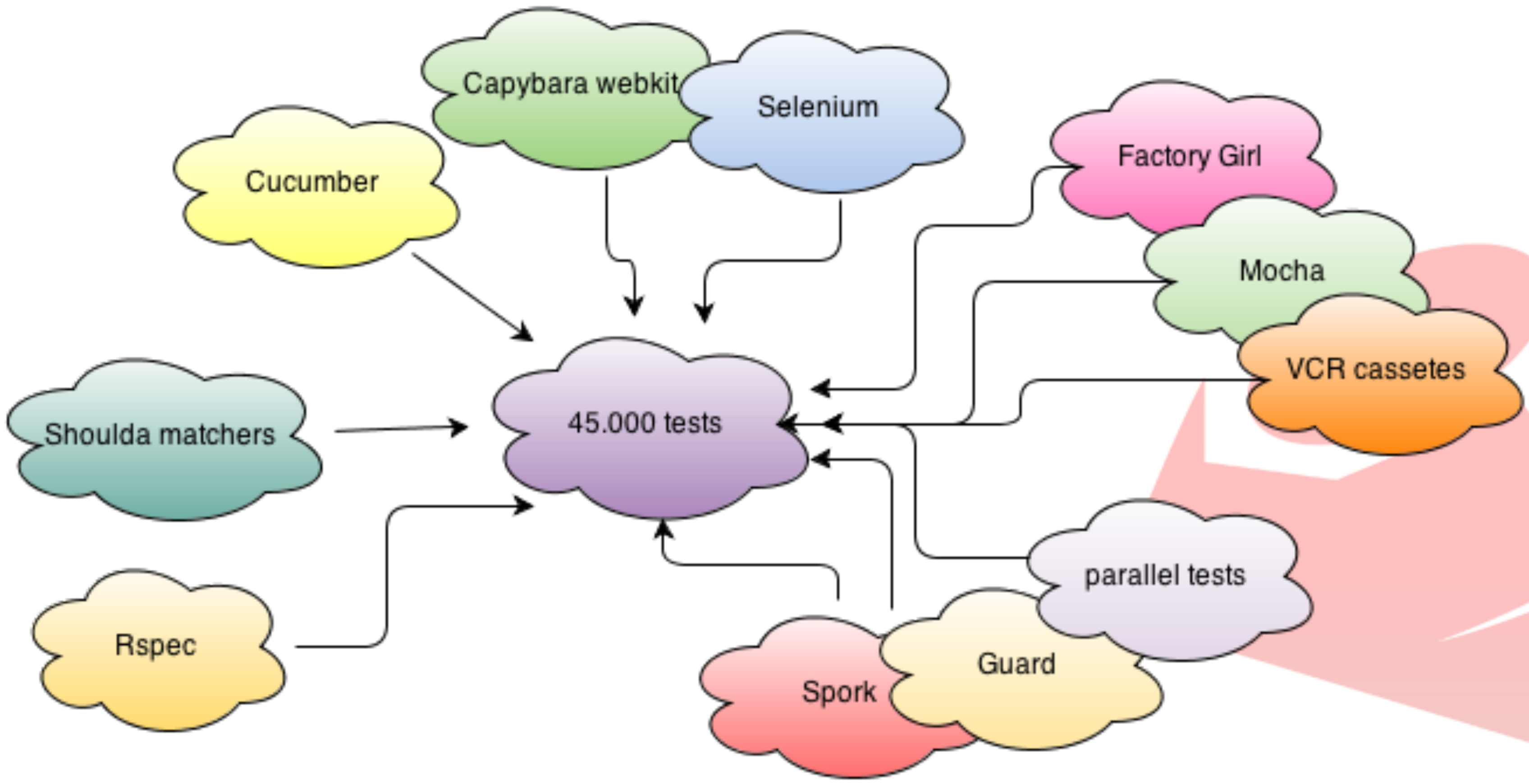
- 35.000 tests back in April
- Currently it's like 45.000 tests...



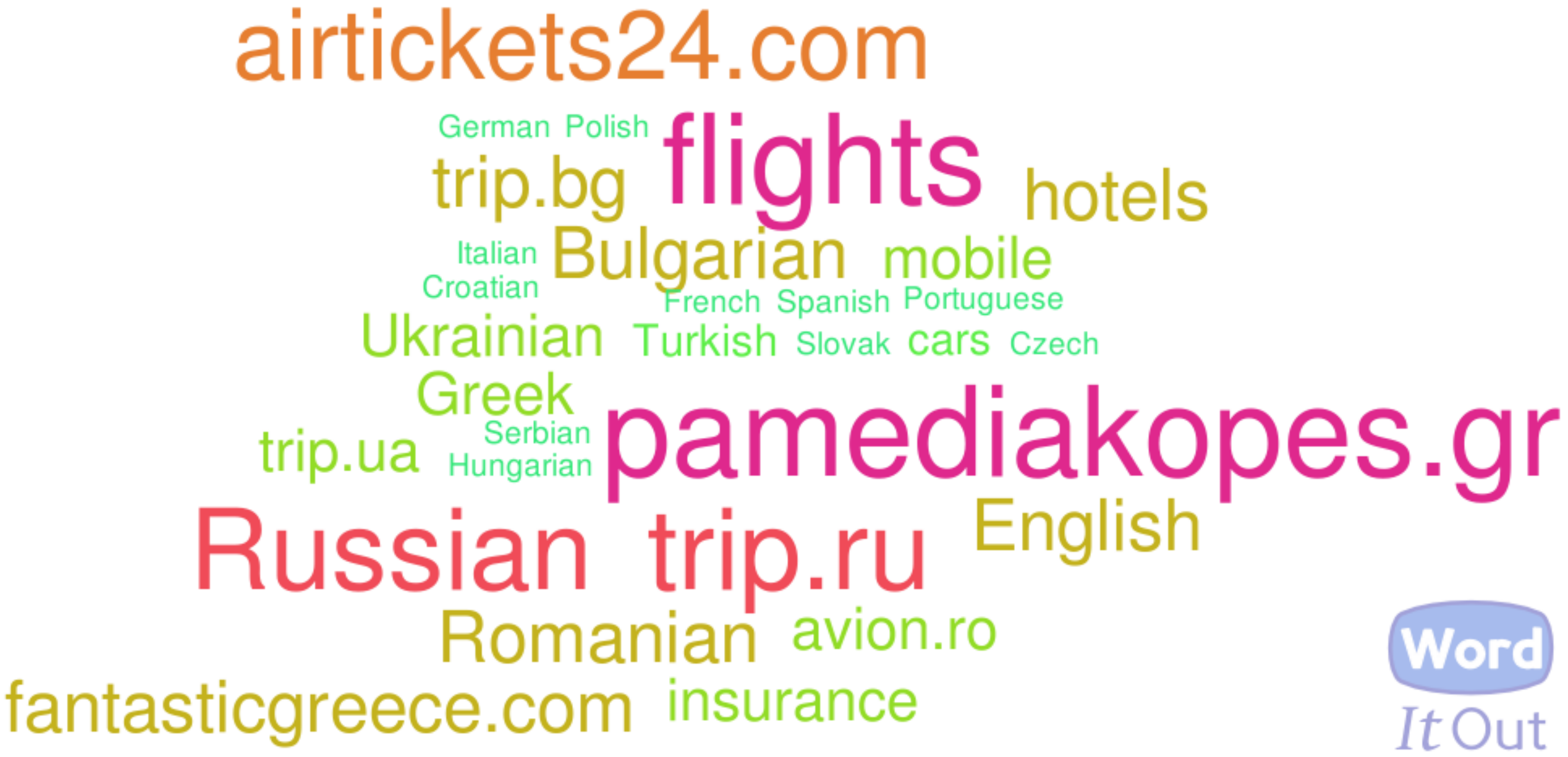
To be more precise



Technologies Used



But... Why so many tests?



Still... Why so many??

- Many visitors and transactions
- Each line of code is considered an investment

How to integrate testing

- No tests = no commits
- Add tests in parallel with development
- If the code is too complicated to test, rewrite it

Development and maintenance

- D-I-S-C-I-P-L-I-N-E !!
- Generalized steps
- Constant clean up and refactorings

Proper Naming

- Spec tests are named after the name of what they describe
  - Example: searches_controller.rb
    - searches_controller_spec.rb

- Feature tests are named after the user story they describe
  - Example: Search form
    - autocomplete.feature
    - validations.feature

Proper Division

- Spec tests are under the same namespace of what they describe
  - Example: searches /new.html.erb
    - searches / new.html.erb_spec.rb

- Feature tests are divided in conceptual groups that cover all the stories of an action
  - Example: Search form
    - search / autocomplete.feature
    - search / validations.feature

Advantages

- Leads to self-explanatory code
- Ease of blending in for new members
- Ease of refactoring
- Safer releases

What's the catch??

- Run time needed
- Maintenance is really tedious and time consuming
- 1 hour of coding = at least 2 hours of testing

Having tests means 100% bug free?

- NO !!
- Why ???
  - Unknown external factors
  - Development != Production
  - Different caching
  - Load balancer, etc
- Holes in initial specs
- Tests can't check what they don't know

Making the procedure faster

- Continuous Integration (CI) server
- Run only the tests you need
- Locally run tests in parallel
- Use of guard and spork

When not to write a test

- For code you don't have
- For tests you already have

Mistakes we learnt from

- Wrote too many feature tests
- Tested the same thing over and over
- Used solely selenium-webdriver
- VCR recordings

Future todo's

- Run tests in parallel on the CI server
- Add more agents to be able to run more configurations in parallel
- Finish clean ups
- More coverage

Conclusion

- Test, test, test and test some more
- Tests are not a panacea
- Write specs for non functional tests
- Set a structure and follow it. It will reward you

Questions ??