



Мутационное тестирование

STRYKER.JS

Варпахович Кирилл Николаевич

Ведущий FrontEnd Разработчик

Tinkoff.ru

О чём поговорим?



- Уровни тестирования
- Взросление продукта
- Взросление бизнеса
- Что такое мутационное тестирование
- Как начать работать с Stryker
- Подводные камни, которые мы поймали



Типы тестирования

* по уровню

Типы тестирования (по уровню)

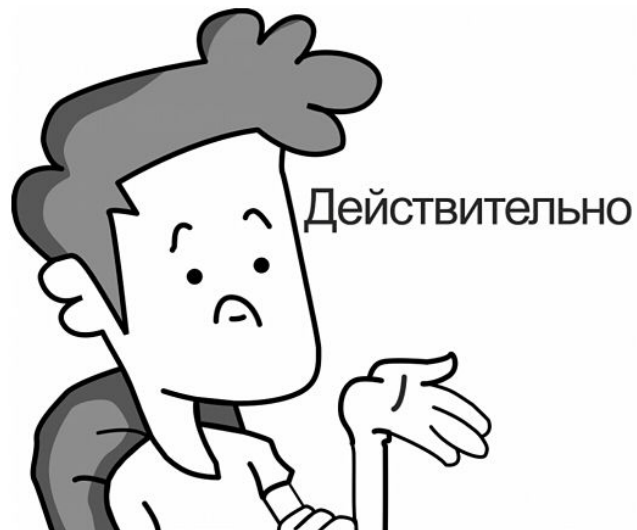


- Как вы тестируете ваше приложение?
- Мы пишем юнит тесты.
- Какие?
- Я же сказал - юнит!

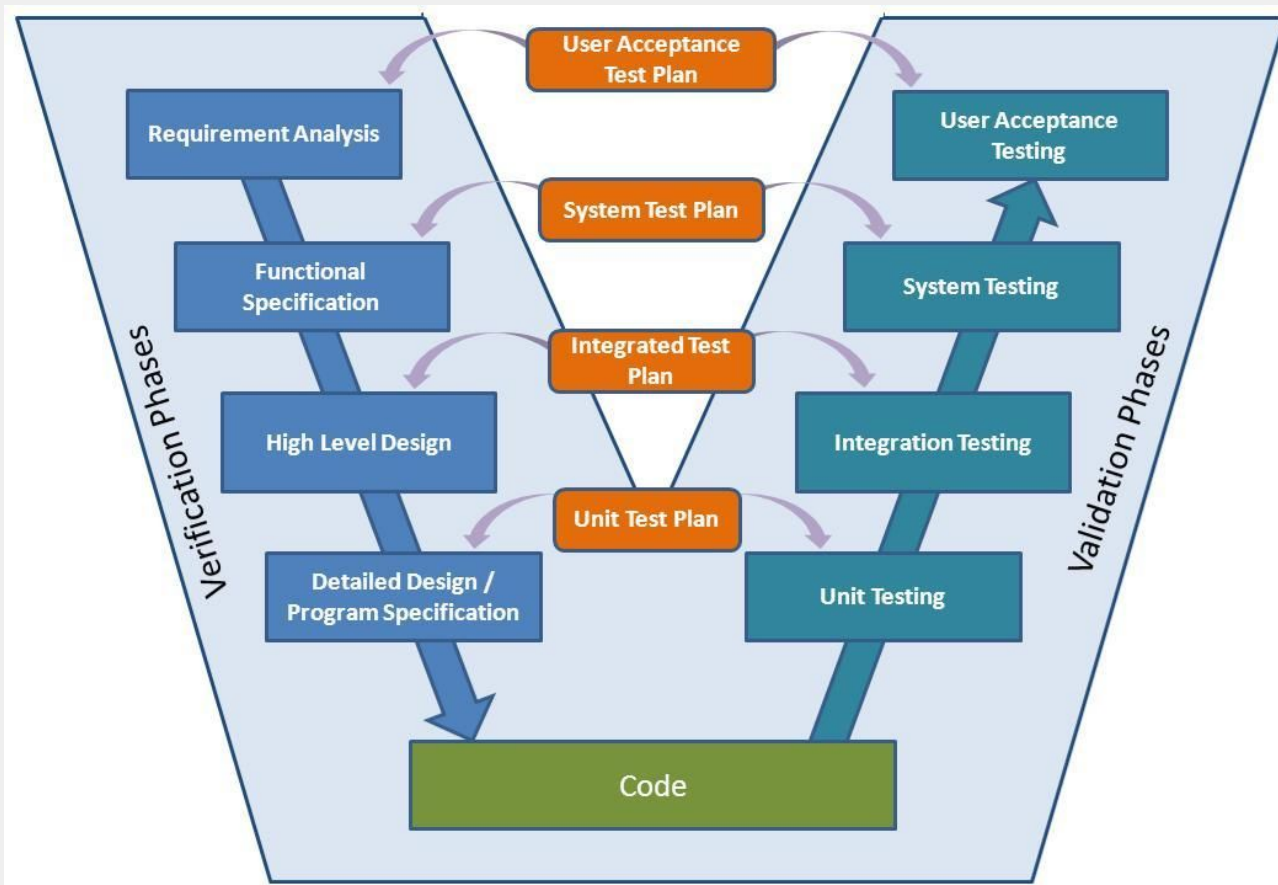
Типы тестирования (по уровню)



- Как вы тестируете ваше приложение?
- Мы пишем юнит тесты.
- Какие?
- Я же сказал - юнит!



Типы тестирования (по уровню)



Типы тестирования (по уровню)



Системное тестирование
(сценарии использования и т.д.)

Системное тестирование

- проверяют выполнение бизнес-функций (требований) системы.
- проверяет сценарии работы пользователя в системе
- есть много видов

Типы тестирования (по уровню)



Системное тестирование
(сценарии использования и т.д.)

Тестирование интеграционное
(проверяем результат внешних эффектов)

Тестирование интеграционное

- проверяем изменения после взаимодействия компонент
- проверяем технические сценарии взаимодействия компонент

Типы тестирования (по уровню)



Тестирование компоненты

- Проверяем что вызываются внешние функции/сущности
- Проверяем обработку запросов на изменение извне
- Дёргаем внешнее API компоненты

Типы тестирования (по уровню)



Тестирование модулей

- Проверяем нечистые функции
- Проверяем функции, которые должны менять состояние модуля
- Проверяем, что чистые функции не меняют состояние модуля

Типы тестирования (по уровню)



Тестирование функций

- Проверяем результат чистых функций
- Проверяем результат не чистых функций на разных входных параметрах
- Не проверяем изменение состояния сущности

Типы тестирования (по уровню)



Это всё
Unit-тестирование



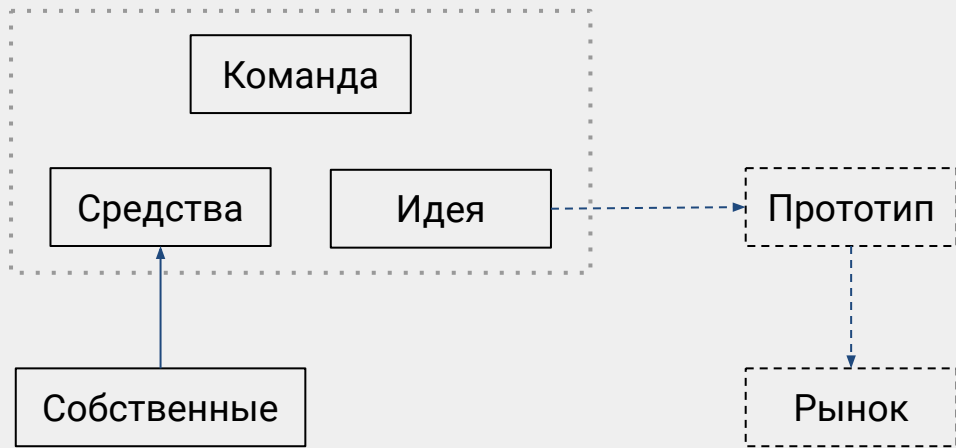
Когда тесты нужны бизнесу?

Этапы взросления

Взросление бизнеса



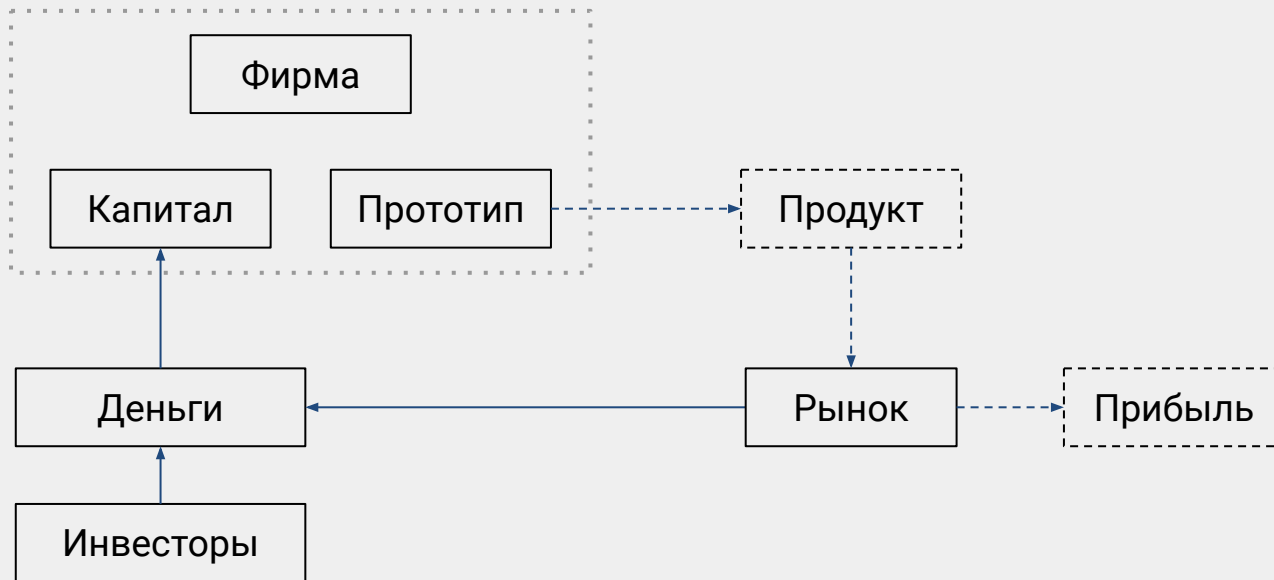
Этап 1



Взросление бизнеса



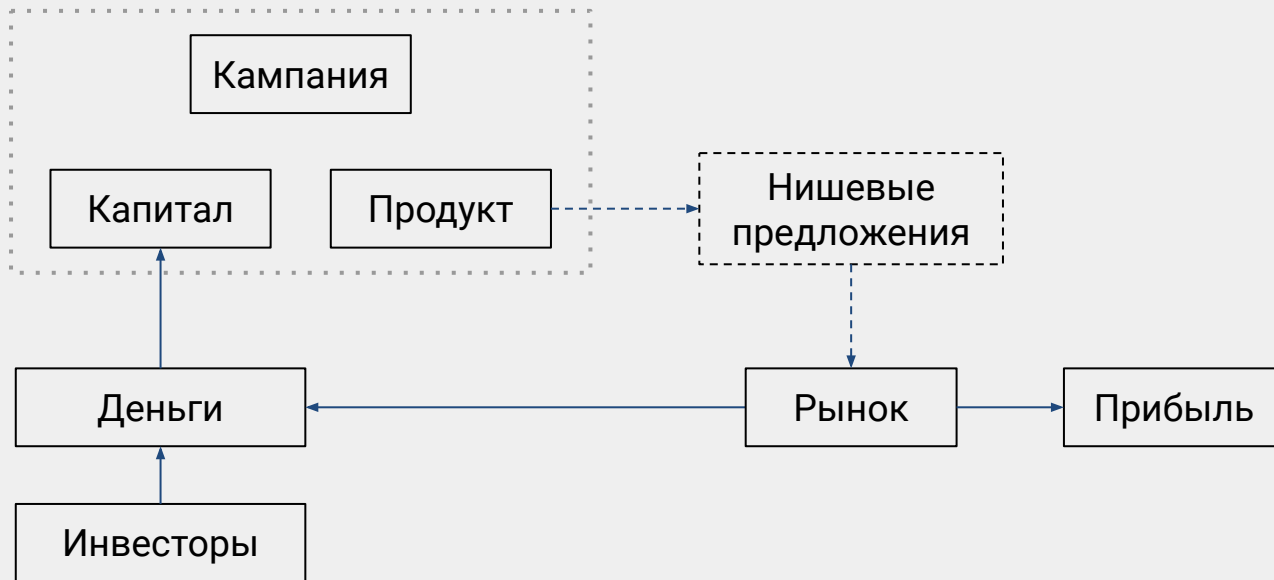
Этап 2



Взросление бизнеса



Этап 3



Взрождение продукта



Специальный продукт

Укоренение в рынке
Обеспечение стабильности

Работоспособный продукт
Market Product

Демонстрация жизнеспособности
Максимизация прибыли

Опытный образец
Minimal viable product

Демонстрация возможностей

Прототип
Proof of concept

Проверка осуществимости

Идея
Concept

Проверка востребованности

Взрождение продукта



Специальный продукт

Укоренение в рынке
Обеспечение стабильности

Работоспособный продукт
Market Product

Демонстрация жизнеспособности
Максимизация прибыли

Опытный образец
Minimal viable product

Демонстрация возможностей

Прототип
Proof of concept

Проверка осуществимости

Идея
Concept

Проверка востребованности

Взрождение продукта



Специальный продукт

Укоренение в рынке
Обеспечение стабильности

Работоспособный продукт
Market Product

Демонстрация жизнеспособности
Максимизация прибыли

Опытный образец
Minimal viable product

Демонстрация возможностей

Прототип
Proof of concept

Проверка осуществимости

Идея
Concept

Проверка востребованности

Взрождение продукта



Специальный продукт

Укоренение в рынке
Обеспечение стабильности

Работоспособный продукт
Market Product

Демонстрация жизнеспособности
Максимизация прибыли

Опытный образец
Minimal viable product

Демонстрация возможностей

Прототип
Proof of concept

Проверка осуществимости

Идея
Concept

Проверка востребованности

Взрождение продукта



Специальный продукт

Укоренение в рынке
Обеспечение стабильности

Работоспособный продукт
Market Product

Демонстрация жизнеспособности
Максимизация прибыли

Опытный образец
Minimal viable product

Демонстрация возможностей

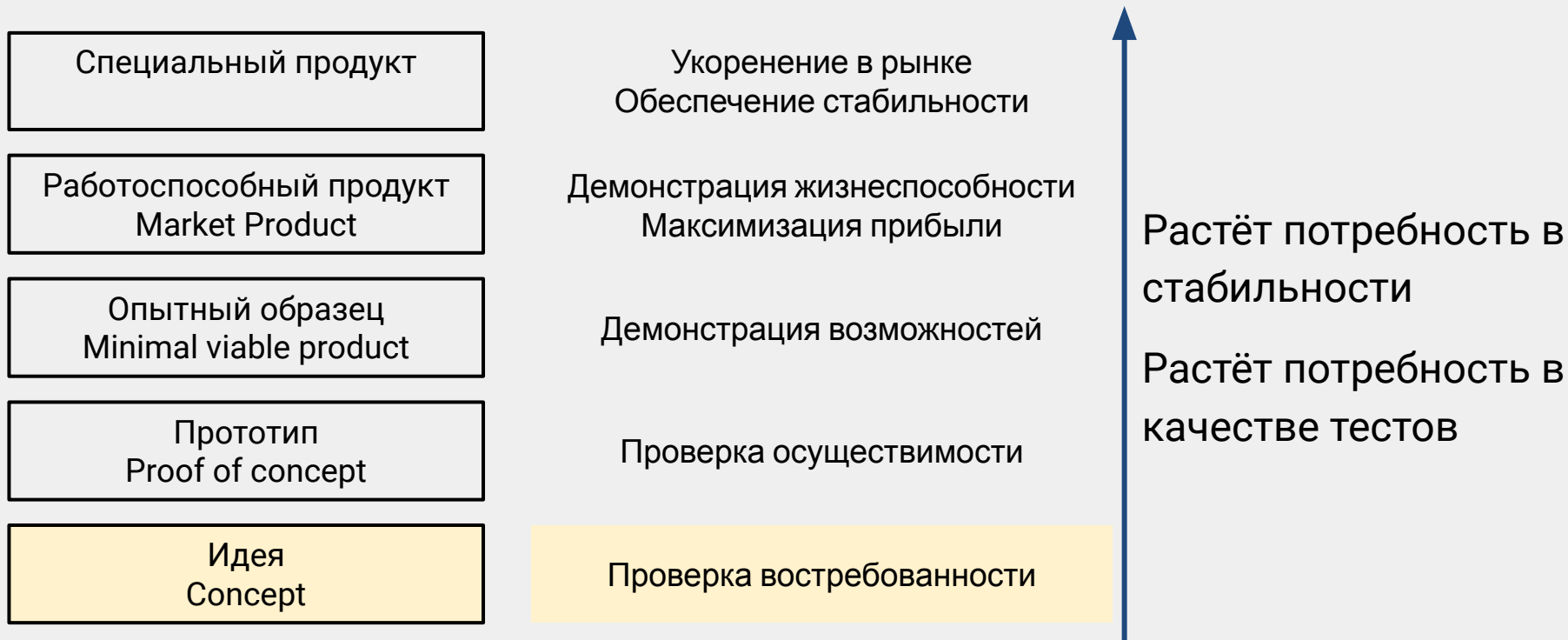
Прототип
Proof of concept

Проверка осуществимости

Идея
Concept

Проверка востребованности

Взрождение продукта





Бизнес готов
Тесты пишутся
Что дальше?

Качество тестирования



Идея
Concept

Проверка востребованности

Прототип
Proof of concept

Проверка осуществимости

Опытный образец
Minimal viable product

Демонстрация возможностей

Работоспособный продукт
Market Product

Демонстрация жизнеспособности
Максимизация прибыли

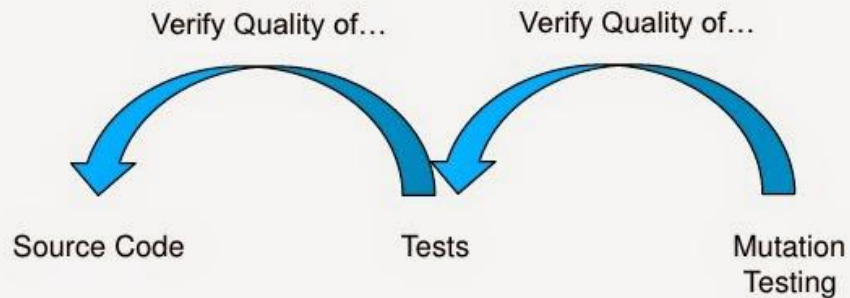
Специальный продукт

Укоренение в рынке
Обеспечение стабильности

Растёт потребность в
стабильности

Растёт потребность в
качестве тестов

What is Mutation Testing?

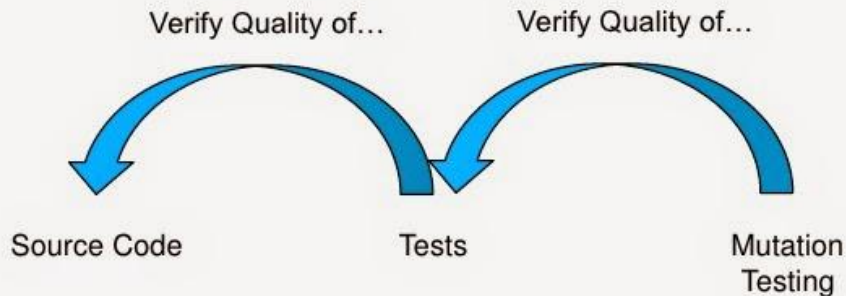




Цели:

- Повышаем качество тестов
- Удаляем бесполезные тесты
- Повышаем качество кода
- Попутно улучшаем архитектуру, отказоустойчивость и т.д.

What is Mutation Testing?



Качество тестирования



Stryker:

C#

Scala

Javascript/Typescript

React

Angular

Native



Stryker:

C#

Scala

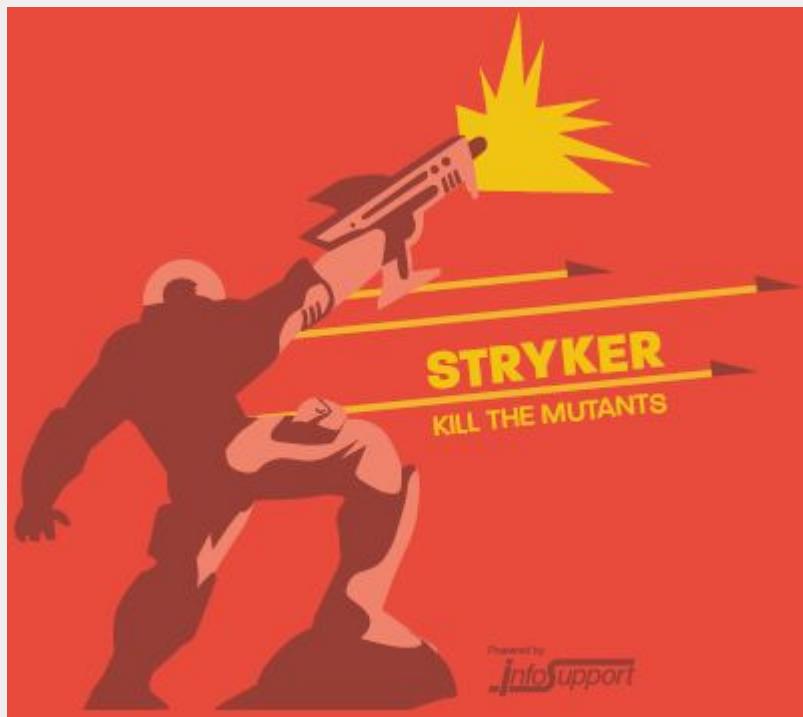
Javascript/Typescript

React

Angular

Native

Качество тестирования





Getting started...

(немного чёрных экранов)



```
npm install -g stryker-cli
```

```
npm install -D stryker-cli
```



```
npm install -g stryker-cli
```

```
npm install -g stryker-cli
```

```
npm install --save-dev @stryker-mutator/core
```

```
npm install --save-dev @stryker-mutator/html-reporter  
npm install --save-dev @stryker-mutator/typescript  
npm install --save-dev @stryker-mutator/jest-runner
```




```
npm install -g stryker-cli
```

```
npm install -g stryker-cli
```

```
npm install --save-dev @stryker-mutator/core
```

```
npm install --save-dev @stryker-mutator/html-reporter  
npm install --save-dev @stryker-mutator/typescript  
npm install --save-dev @stryker-mutator/jest-runner
```

```
stryker init
```



A diagram showing a large, light-yellow rectangular area representing a code block. The text "stryker run" is centered within this area. To the left of the block, there are three small, vertically stacked rectangular boxes containing the letters "n", "n", and "s" respectively. To the right of the block, there are two small, vertically stacked rectangular boxes. Above the block, there are two horizontal lines that extend from the left and right edges of the block, suggesting a container or a specific execution context.

stryker run





stryker.conf.js

```
module.exports = function(config) {  
  config.set({  
    ...  
  });  
};
```



stryker.conf.js

```
module.exports = function(config) {
  config.set({
    ...
    mutator: "typescript",
    testRunner: "jest",
    jest: {
      configFile: "jest.conf.js",
      projectName: "angular-cli",
    },
  });
};
```



stryker.conf.js

```
module.exports = function(config) {
  config.set({
    ...
    reporters: [
      "progress",
      "clear-text",
      "html"
    ],
    htmlReporter: {
      baseDir: 'reports/mutation/html'
    },
  });
};
```



stryker.conf.js

```
module.exports = function(config) {  
  config.set({  
    ...  
    maxConcurrentTestRunners: 4,  
    // Recommended to use about half of your available cores when running stryker with angular.  
  });  
};
```

stryker.conf.js

```
module.exports = function(config) {  
  config.set({  
    ...  
    maxConcurrentTestRunners: 4  
    // Recommended to use about  
  });  
};
```



stryker with angular.



stryker.conf.js

```
module.exports = function(config) {  
  config.set({  
    mutate: [  
      "src/**/*.ts",  
      "!src/**/*.spec.ts",  
      "!src/**/*.module.ts",  
      "!src/app/test/**/*.ts",  
      "!src/**/*.routing.ts"  
    ],  
  });  
};
```



```
> stryker run  
  
> INFO ConfigReader Using stryker.conf.js in the current working directory.  
> INFO InputFileResolver Found 992 of 2002 file(s) to be mutated.  
> INFO InitialTestExecutor Starting initial test run. This may take a while.  
> INFO InitialTestExecutor Initial test run succeeded. Ran 1170 tests in 2 minutes 45  
seconds (net 54338 ms, overhead 65737 ms).  
> INFO MutatorFacade 12094 Mutant(s) generated  
> INFO SandboxPool Creating 4 test runners (based on maxConcurrentTestRunners config)
```



```
> stryker run  
  
> INFO ConfigReader Using stryker.conf.js in the current working directory.  
> INFO InputFileResolver Found 992 of 2002 file(s) to be mutated.  
> INFO InitialTestExecutor Starting initial test run. This may take a while.  
> INFO InitialTestExecutor Initial test run succeeded. Ran 1170 tests in 2 minutes 45  
seconds (net 54338 ms, overhead 65737 ms).  
> INFO MutatorFacade 12094 Mutant(s) generated  
> INFO SandboxPool Creating 4 test runners (based on maxConcurrentTestRunners config)
```





```
> stryker run

> INFO ConfigReader Using stryker.conf.js in the current working directory.
> INFO InputFileResolver Found 992 of 2002 file(s) to be mutated.
> INFO InitialTestExecutor Starting initial test run. This may take a while.
> INFO InitialTestExecutor Initial test run succeeded. Ran 1170 tests in 2 minutes 45 seconds (net
54338 ms, overhead 65737 ms).
> INFO MutatorFacade 12094 Mutant(s) generated
> INFO SandboxPool Creating 4 test runners (based on maxConcurrentTestRunners config)

-----|-----|-----|-----|-----|-----|-----|
File      | % score | # killed | # timeout | # survived | # no cov | # error |
-----|-----|-----|-----|-----|-----|-----|
All files |    0.00 |         0 |          0 |        12094 |          0 |         0 |
  call-step-form.component.ts |    0.00 |         0 |          0 |          11 |          0 |         0 |
  ...     |    0.00 |         0 |          0 |          ... |          0 |         0 |
-----|-----|-----|-----|-----|-----|

> INFO HtmlReporter Your report can be found at: file:///.../reports/mutation/html/index.html
> INFO Stryker Done in 6 hours 11 minutes.
```



Done in 6 hours 11 minutes

<https://github.com/stryker-mutator/stryker-handbook/blob/master/mutant-states-and-metrics.md>

File	% score	# killed	# timeout	# survived	# no cov	# error
All files	0.00	0	0	12094	0	0
call-step-form.component.ts	0.00	0	0	11	0	0
...	0.00	0	0	...	0	0

перцентил
ь

циферь

умерло от безысходности

живее всех живых

этих даже не искали

эти поломали нам
систему


```
htmlReporter: { baseDir: 'reports/mutation/html' },
```

All files



File / Directory	Mutation score	# Killed	# Survived	# Timeout	# No coverage	# Runtime errors	# Compile errors	Total detected	Total undetected	Total mutants
📁 All files	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	361	0	0	0	0	361	361	
📁 abstract	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	129	0	0	0	0	129	129	
📄 first/first-step.helper.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	9	0	0	0	0	9	9	
📄 second/second-step.helper.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	93	0	0	0	0	93	93	
📄 third/third-step.helper.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	14	0	0	0	0	14	14	
📄 fourth/fourth-step.helper.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	14	0	0	0	0	14	14	
📄 fifth/fifth-step.helper.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	12	0	0	0	0	12	12	
📄 sixth/sixth-step.helper.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	9	0	0	0	0	9	9	
📄 another/another.helper.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	71	0	0	0	0	71	71	
📄 step-helper-by-type.ts	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0.00%	0	10	0	0	0	0	10	10	


```
htmlReporter: { baseDir: 'reports/mutation/html' },
```

✓  **Survived (10)** Expand all

```
*/
259 260 templateSettings.interpolate = /{{([\s\S]+?)}}/g;

/**
 * Список шаблонов ошибок компонента
 */
const NO_COMPONENT_FOR_STEP_TYPE = template(261 'Для типа {{type}} в stepHelperByType отсутствует хелпер');

type StepHelper = AbstractStepHelper<AbstractStepData, AbstractStepForm>;


const formHelpers: Record<StepTypes, StepHelper> = 262 {
  [StepTypes.FIRST]: FirstStepHelper.Instance,
  [StepTypes.SECOND]: SecondStepHelper.Instance,
  [StepTypes.THIRD]: ThirdStepHelper.Instance,
  [StepTypes.FOURTH]: FourthStepHelper.Instance,
  [StepTypes.FIFTH]: FifthStepHelper.Instance,
  [StepTypes.SIXTH]: SixthStepHelper.Instance,
};

export function stepHelperByType(type: StepTypes): StepHelper 263 {} {
  const formHelper: StepHelper = formHelpers[type];

  if (264 265 266 !formHelper) 267 {
    throw new Error(NO_COMPONENT_FOR_STEP_TYPE(2
  )

  return formHelper;
}
```

Block

 Survived



Особенности

Особенности



- Очень много времени на анализ результатов
 - Можно настроить запуск на конкретные файлы вручную

Особенности



- Очень много времени на анализ результатов
 - Можно настроить запуск на конкретные файлы вручную
- 1 мутация != 1 прогон
 - :(

Особенности



- Очень много времени на создание тестов
 - Можно настроить
- 1 мутация != 1 прог
 - :(

Arithmetic Operator	✓
Array Declaration	✓
Assignment Expression	✗
Block Statement	✓
Boolean Literal	✓
Conditional Expression	✓
Equality Operator	✓
Logical Operator	✓
Method Expression	✗
String Literal	✓
Unary Operator	✓
Update Operator	✓

или вручную

Особенности



- Очень много времени на анализ результатов
 - Можно настроить запуск на конкретные файлы вручную
- 1 мутация !== 1 прогон
 - :(
- 1 точка в коде почти всегда 1 мутация
 - Для условий и математики количество мутаций больше
 - Можно создавать кастомные мутации + количество инстансов должно решить проблему

Особенности



- Очень много времени на анализ результатов
 - Можно настроить запуск на конкретные файлы вручную
- 1 мутация !== 1 прогон
 - :(
- 1 точка в коде почти всегда 1 мутация
 - Для условий и математики количество мутаций больше
 - Можно создавать кастомные мутации + количество инстансов должно решить проблему
- Нет teamcity-reporter для интеграции
 - Можно пользоваться репортером jest/karma



Первые результаты



- Стали думать что тестируем и как тестируем
- Появились тесты убивающие мутантов
- Избавляемся от тяжёлых тестов
- Выстраиваем пирамиду тестирования здорового программиста

Спасибо за внимание!



Литература по теме

1. medium.com/@ClrMobile - A Beginner's Guide: POC vs. MVP vs. Prototype
2. geteasyqa.com - Типы тестирования программного обеспечения
3. habr.com - Тестирование. Фундаментальная теория
4. [Инженерное предпринимательство курс Хачин В. Н.](#)
5. https://ru.wikipedia.org/wiki/Системное_тестирование
6. habr.com - Мутационное тестирование
7. https://ru.wikipedia.org/wiki/Мутационное_тестирование
8. <https://stryker-mutator.io>
9. [stryker-mutator - Mutant states and metrics](#)
10. [stryker-mutator - Supported mutators](#)