

SPB  
DOT  
NET



# Молчанов Николай

Руководитель ЦК .NET  
Альфа-Банк

<http://github.com/kroniak>

email: me@kroniak.net



# FLURL <https://flurl.io/>

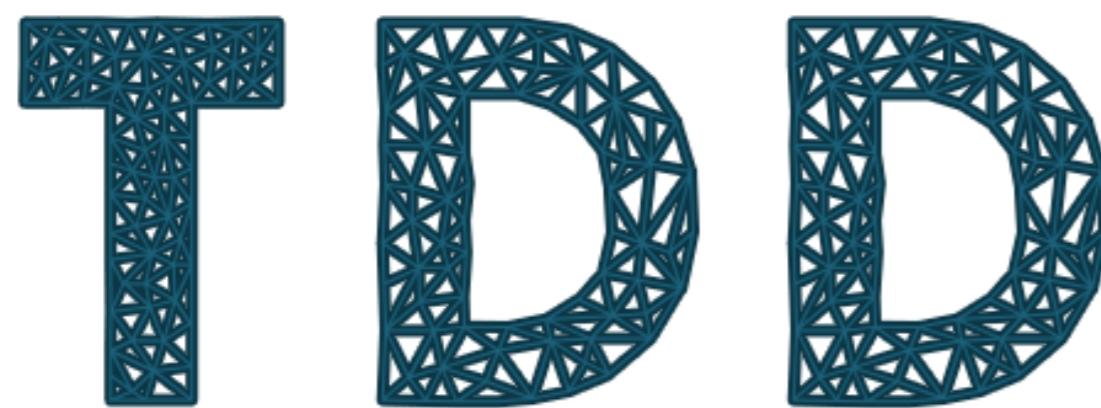
---

```
var result = await "https://api.mysite.com"
    .AppendPathSegment("person")
    .SetQueryParams(new { api_key = "xyz" })
    .WithOAuthBearerToken("my_oauth_token")
    .PostJsonAsync(new { first_name = firstName, last_name = lastName })
    .ReceiveJson<SomeClass>();
```

# FLURL <https://flurl.io/>

---

```
var result = await "https://api.mysite.com"
    .AppendPathSegment("person")
    .SetQueryParams(new { api_key = "xyz" })
    .WithOAuthBearerToken("my_oauth_token")
    .PostJsonAsync(new { first_name = firstName, last_name = lastName })
    .ReceiveJson<SomeClass>();
```



# FLURL <https://flurl.io/>

---

0 references

```
public async Task Test_Some_Http_Calling_MethodAsync()
{
    using (var httpTest = new HttpTest())
    {
        // Arrange

        // Flurl is now in test mode
        await SomeMethod(); // HTTP calls are faked!

        // Assert
    }
}
```

# Test library <https://flurl.io/>

---

0 references

```
public async Task Test_Some_Http_Calling_MethodAsync()
{
    using (var httpTest = new HttpTest())
    {
        httpTest.RespondWithJson(new { x = 1, y = 2 }, 201);

        // Flurl is now in test mode
        await SomeMethod(); // HTTP calls are faked!

        // Assert
    }
}
```

# Test library <https://flurl.io/>

---

0 references

```
public async Task Test_Some_Http_Calling_MethodAsync()
{
    using (var httpTest = new HttpTest())
    {
        httpTest.RespondWithJson(new { x = 1, y = 2 }, 201);

        // Flurl is now in test mode
        await SomeMethod(); // HTTP calls are faked!

        httpTest.ShouldHaveCalled("https://api.mysite.com")
            .WithVerb( HttpMethod.Post )
            .WithContentType("application/json")
            .WithRequestBody("{\"first_name\":\"*", \"last_name\":\"*\"}") // supports wildcards
            .Times(1);
    }
}
```

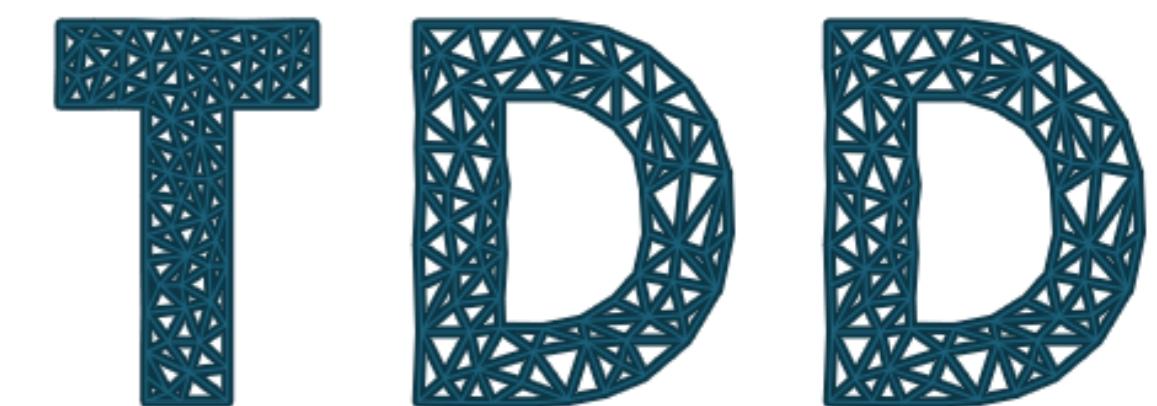
# Ecwid client library

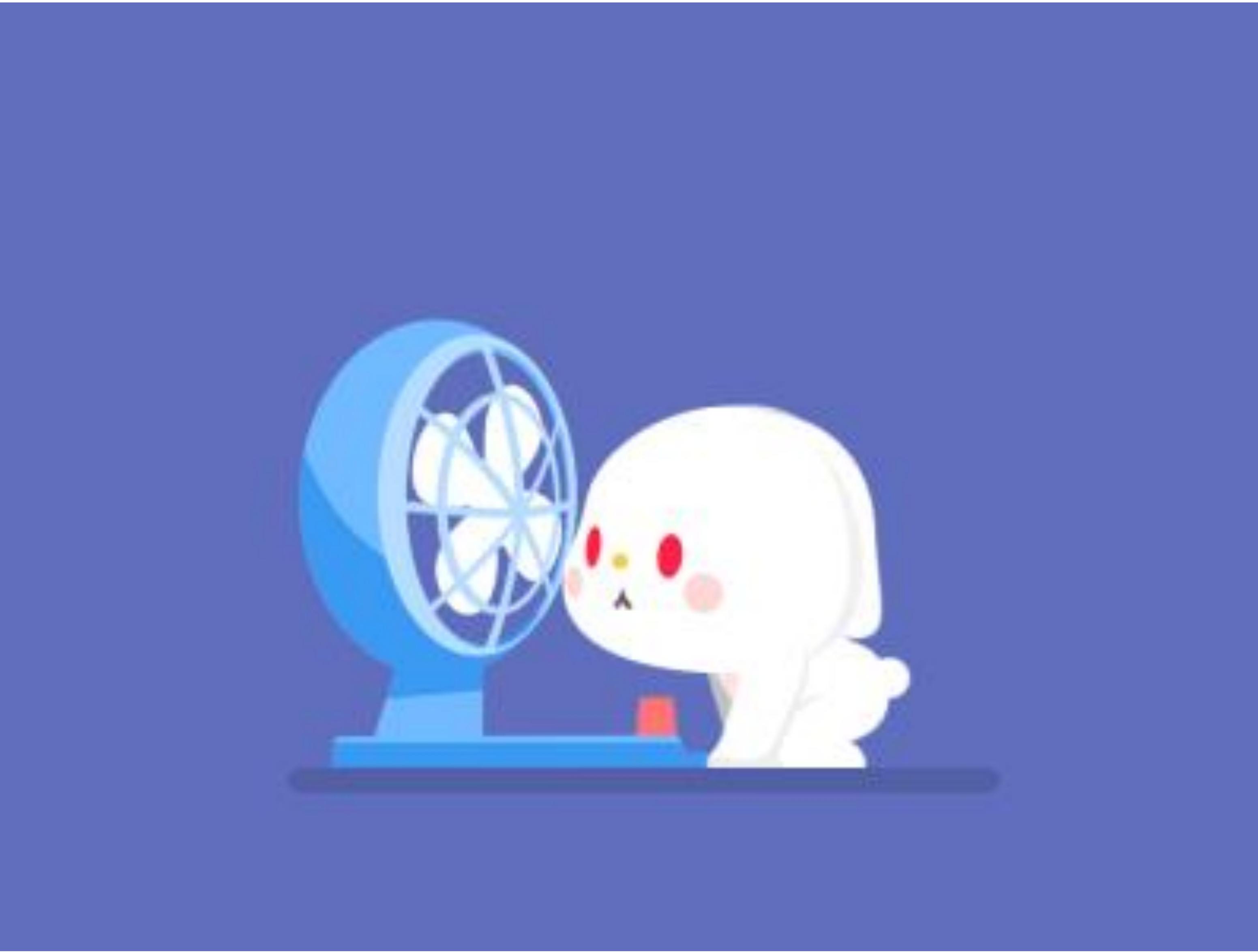
---

```
var result = await new EcwidClient()
    .Configure(someid, "somekey")
    .Orders // get an order query builder for the client
    .PaymentStatuses("PAID") //add some payment statuses
    .FulfillmentStatuses("DELIVERED")
    .CreatedFrom(DateTime.Today) // get orders from this date
    .CreatedTo(DateTime.Now) // and to this date
    .Keywords("cool phone") // set keywords
    .TotalFrom(100.00)
    .TotalTo(200.00) //set totals
    .PaymentMethod("PayPal") // only PayPal
    .Limit(10) // limit one page by this limit
    .Offset(10) // skip 10 orders
    .GetAsync(); // get from API
// and more more filters
```

# Ecwid client library

```
var result = await new EcwidClient()
    .Configure(someid, "somekey")
    .Orders // get an order query builder for the client
    .PaymentStatuses("PAID") //add some payment statuses
    .FulfillmentStatuses("DELIVERED")
    .CreatedFrom(DateTime.Today) // get orders from this date
    .CreatedTo(DateTime.Now) // and to this date
    .Keywords("cool phone") // set keywords
    .TotalFrom(100.00)
    .TotalTo(200.00) //set totals
    .PaymentMethod("PayPal") // only PayPal
    .Limit(10) // limit one page by this limit
    .Offset(10) // skip 10 orders
    .GetAsync(); // get from API
// and more more filters
```





# Мутационное тестирование

# Мутационное тестирование



TECTnpo.a#ne  
My\_valionHoe



# План

---

- тестирование и парадигмы тестирования

# План

---

- тестирование и парадигмы тестирования
- про принцип мутационного тестирования

# План

---

- тестирование и парадигмы тестирования
- про принцип мутационного тестирования
- виды мутантов

# План

---

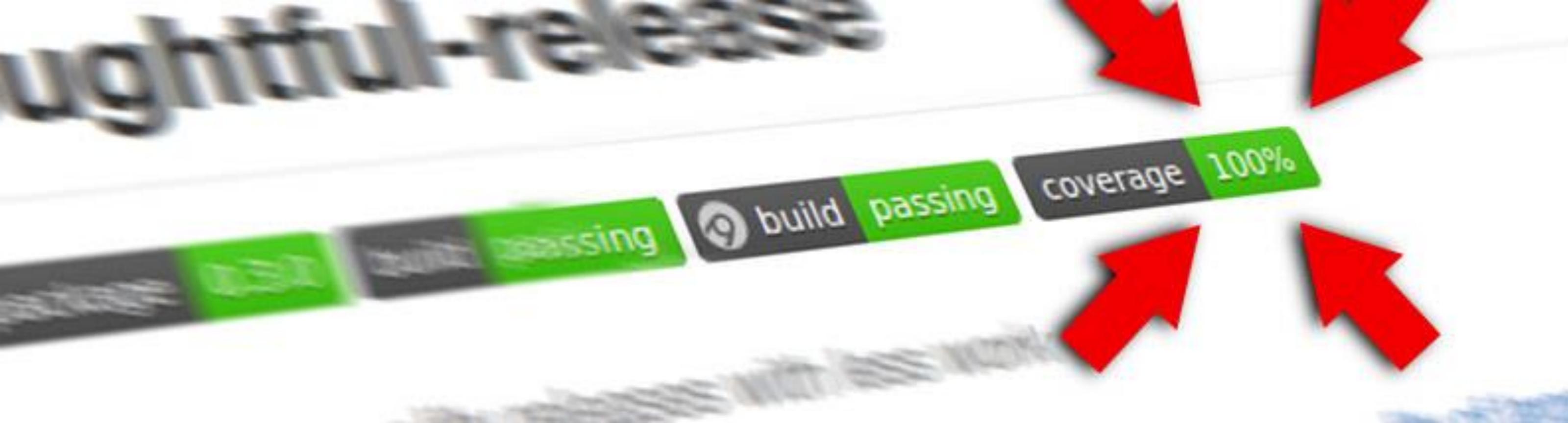
- тестирование и парадигмы тестирования
- про принцип мутационного тестирования
- виды мутантов
- как применить в .NET

# План

---

- тестирование и парадигмы тестирования
- про принцип мутационного тестирования
- виды мутантов
- как применить в .NET
- когда использовать мутационное тестирование

# тестирование

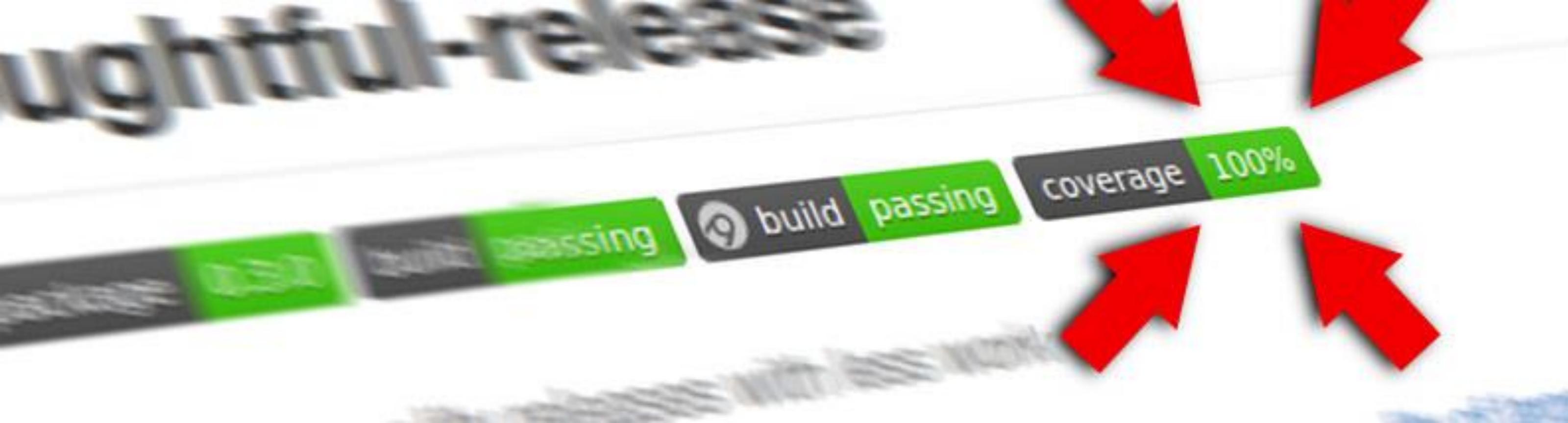


# тестирование

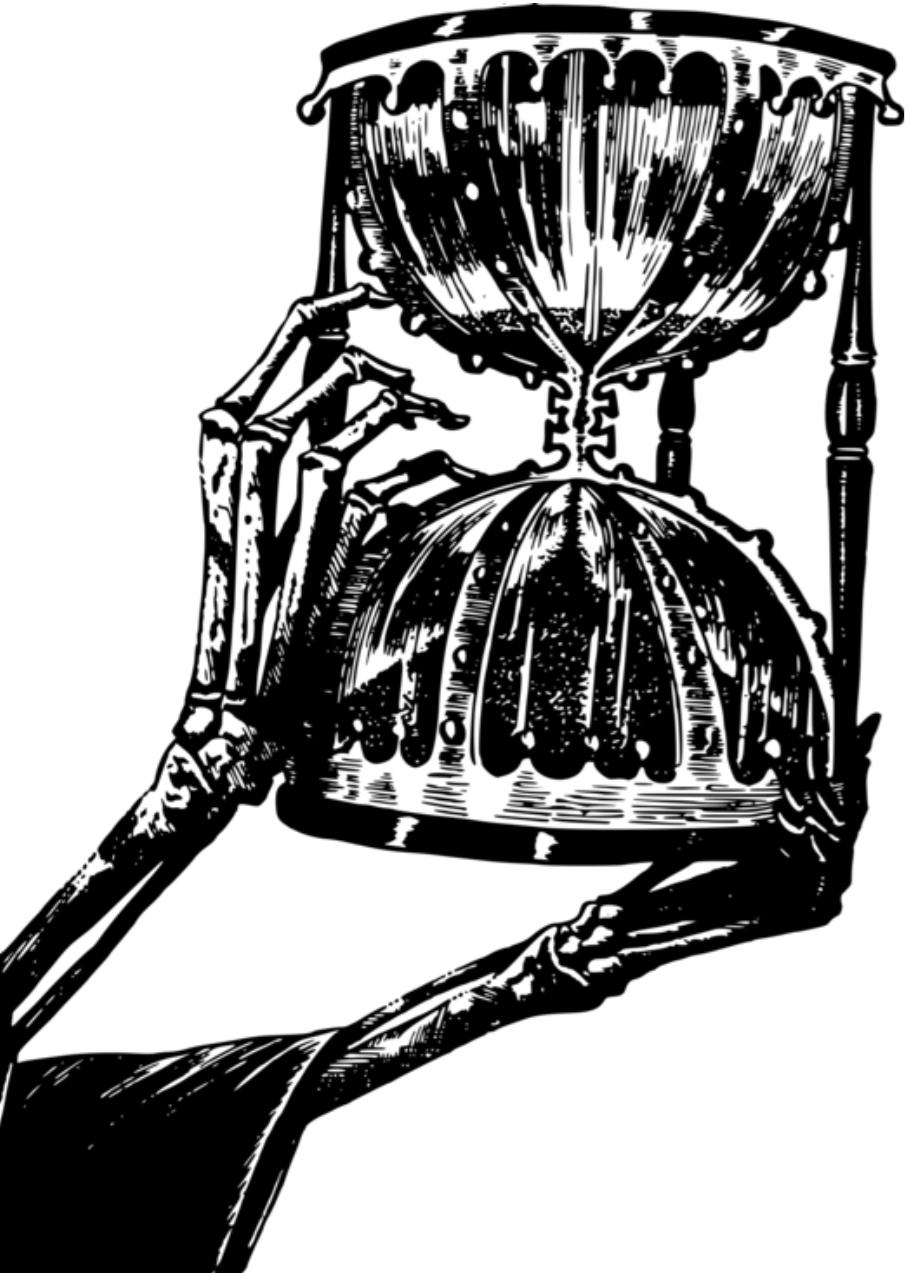


# тестирование

TDD



# тестирование



TDD

Тестирование  
это  
Проверка соответствия  
требованиям



Тестирование  
это  
Проверка соответствия  
требованиям



Покрытие > 70%

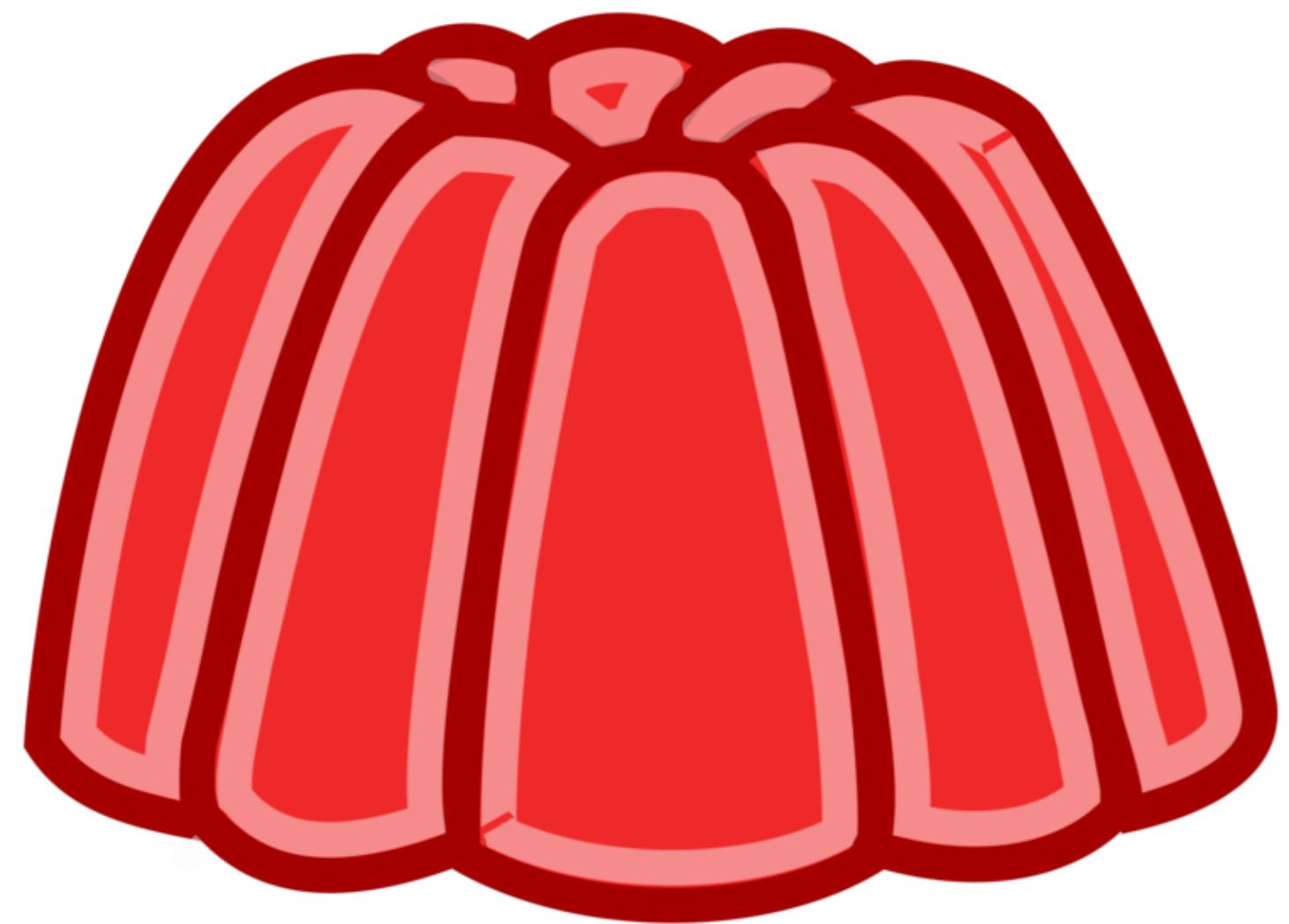


# Oracle

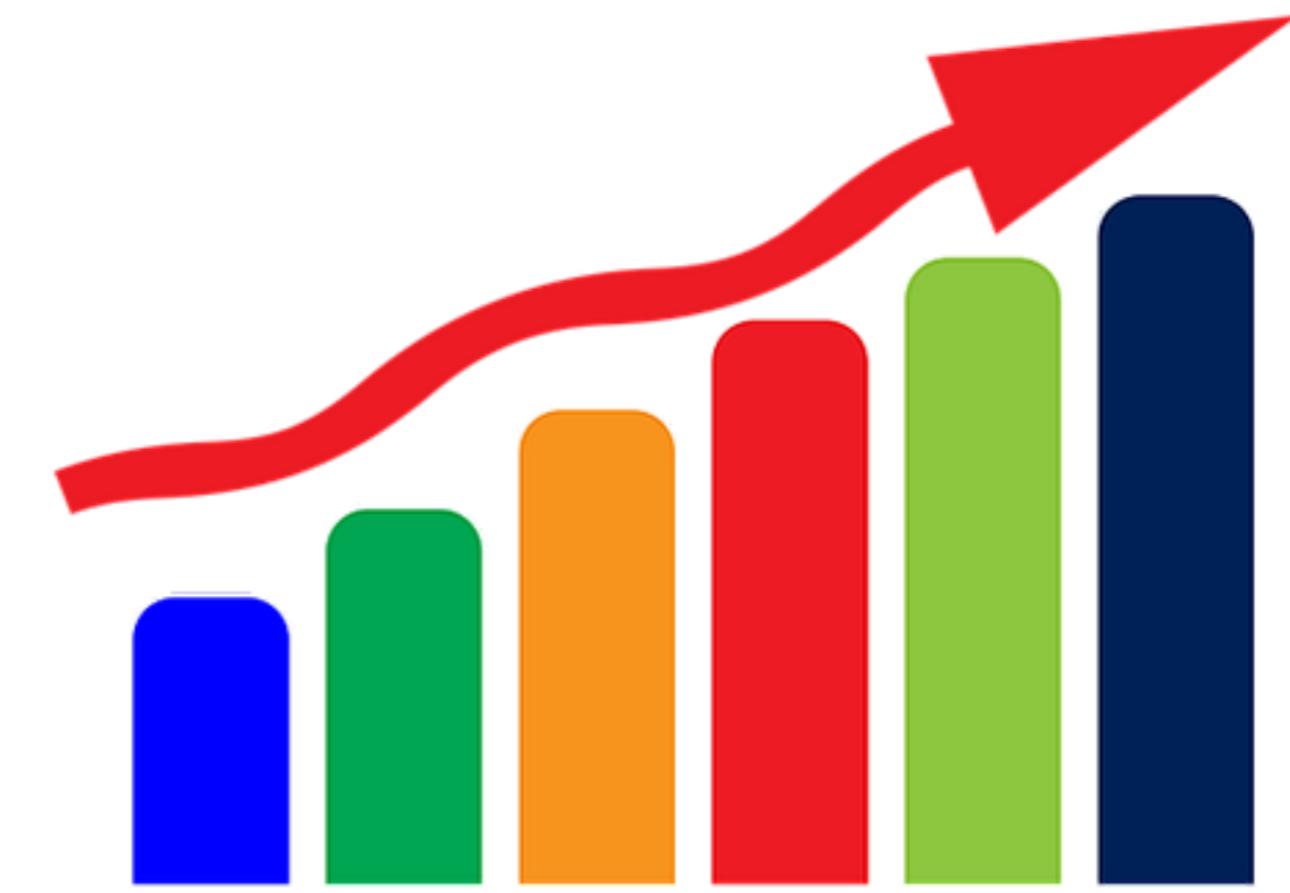
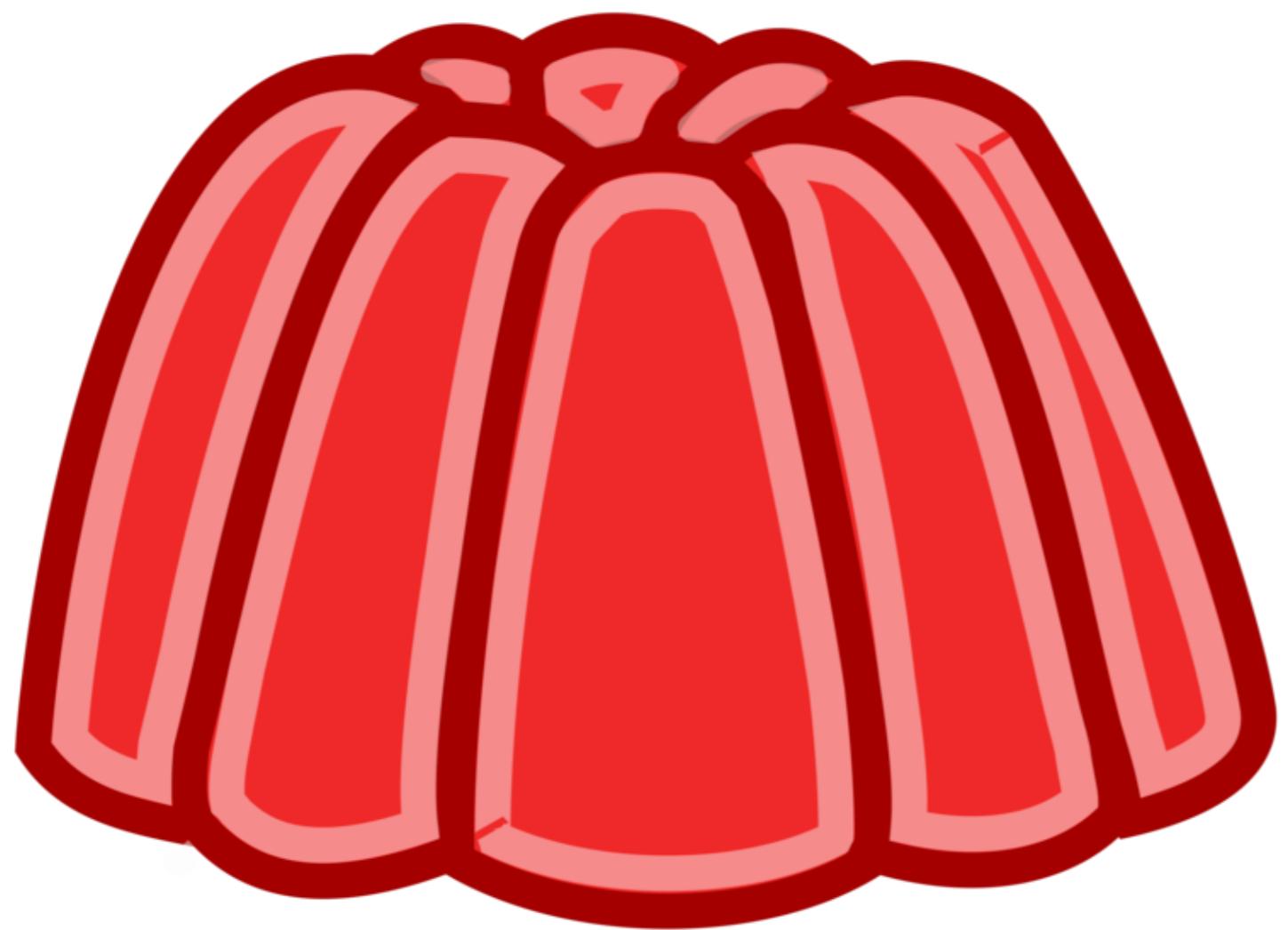
> 40 000 tests

> 5 hour testing

# Oracle



# Oracle



# Как протестировать тесты?

# Как протестировать тесты

---

- code-review кода тестов

# Как протестировать тесты

---

- code-review кода тестов
- code-coverage

# Как протестировать тесты

---

- code-review кода тестов
- code-coverage
- test after each build

# Как протестировать тесты

---

- code-review кода тестов
- code-coverage
- test after each build
- внешний аудит

# Как протестировать тесты

---

- code-review кода тестов
- code-coverage
- test after each build
- внешний аудит
- рандомный набор данных

# Как протестировать тесты

---

- code-review кода тестов
- code-coverage
- test after each build
- внешний аудит
- рандомный набор данных

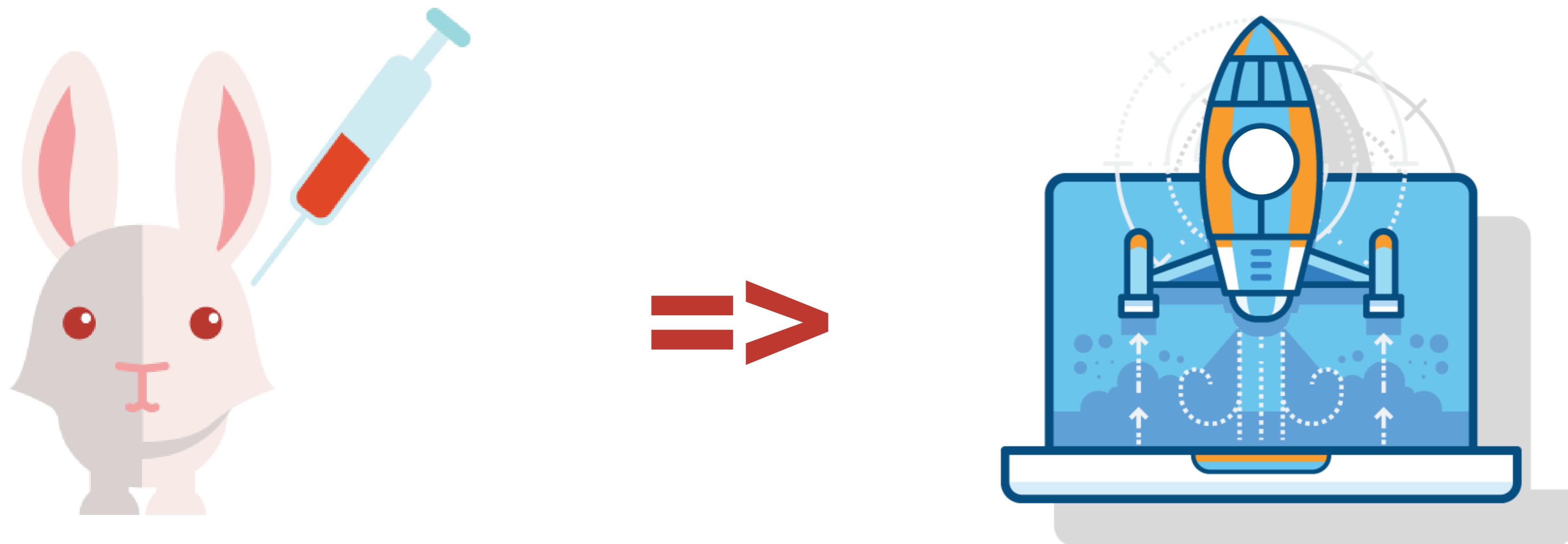


# Как протестировать тесты

---



# Как протестировать тесты



# Парадигмы и виды тестирования

# Виды тестирования

---

- функциональное тестирование

# ви́ды тести́рования

---

- функциональное тестирование
- unit-тестирование

# Виды тестирования

---

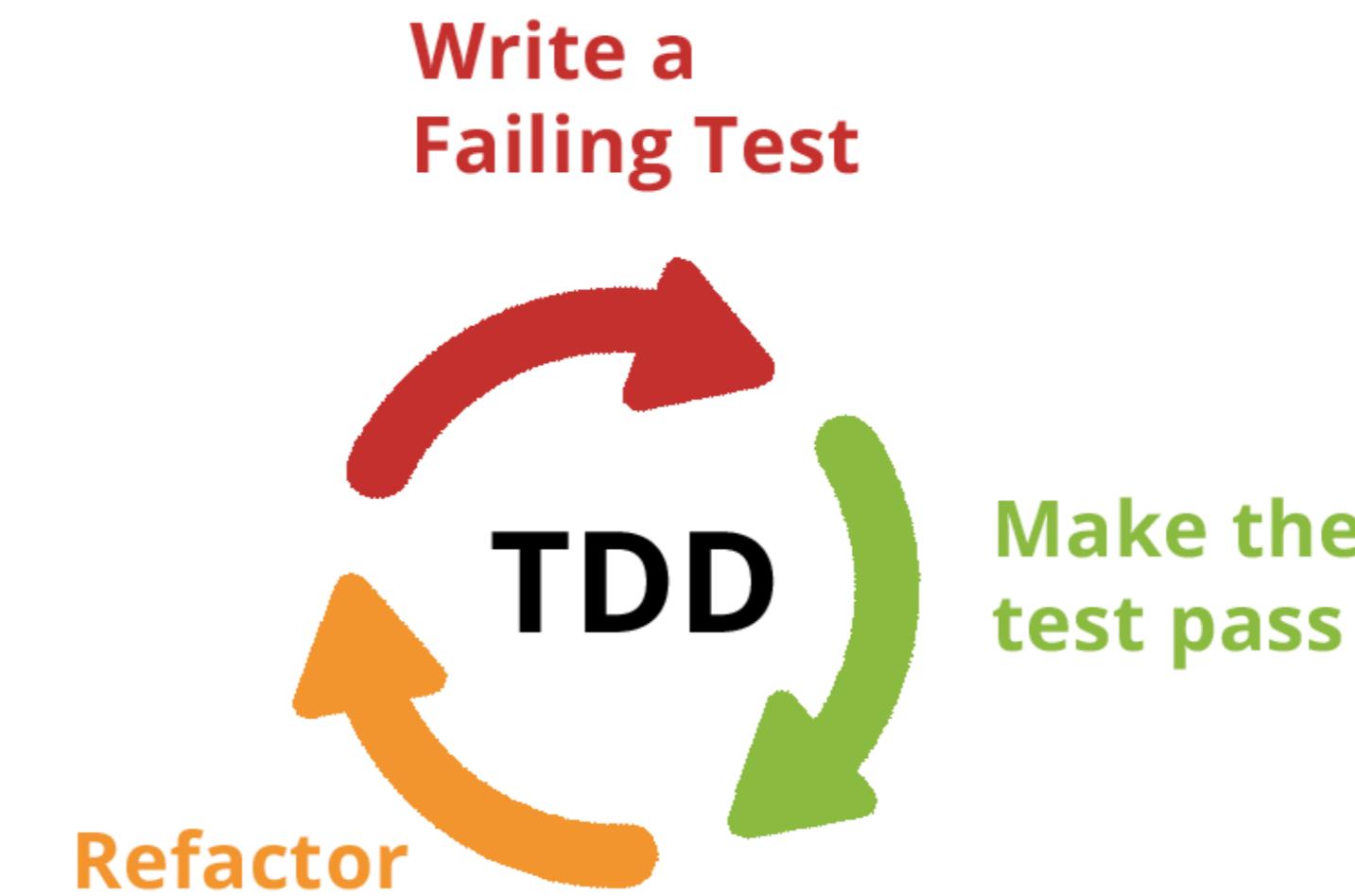
- функциональное тестирование
- unit-тестирование
- автотестирование интеграционное

# Виды тестирования

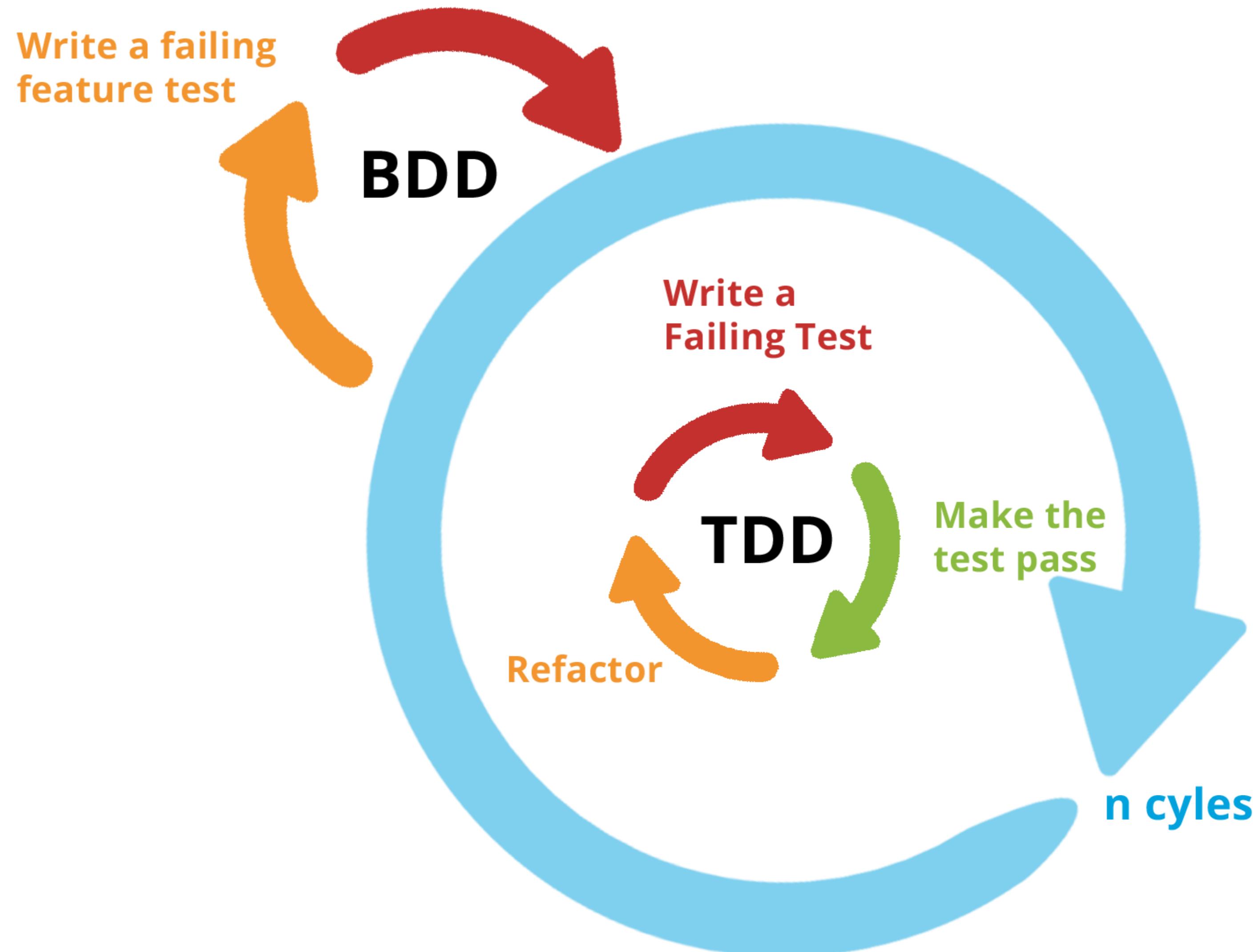
---

- функциональное тестирование
- unit-тестирование
- автотестирование интеграционное
- автотестирование UI и e2e

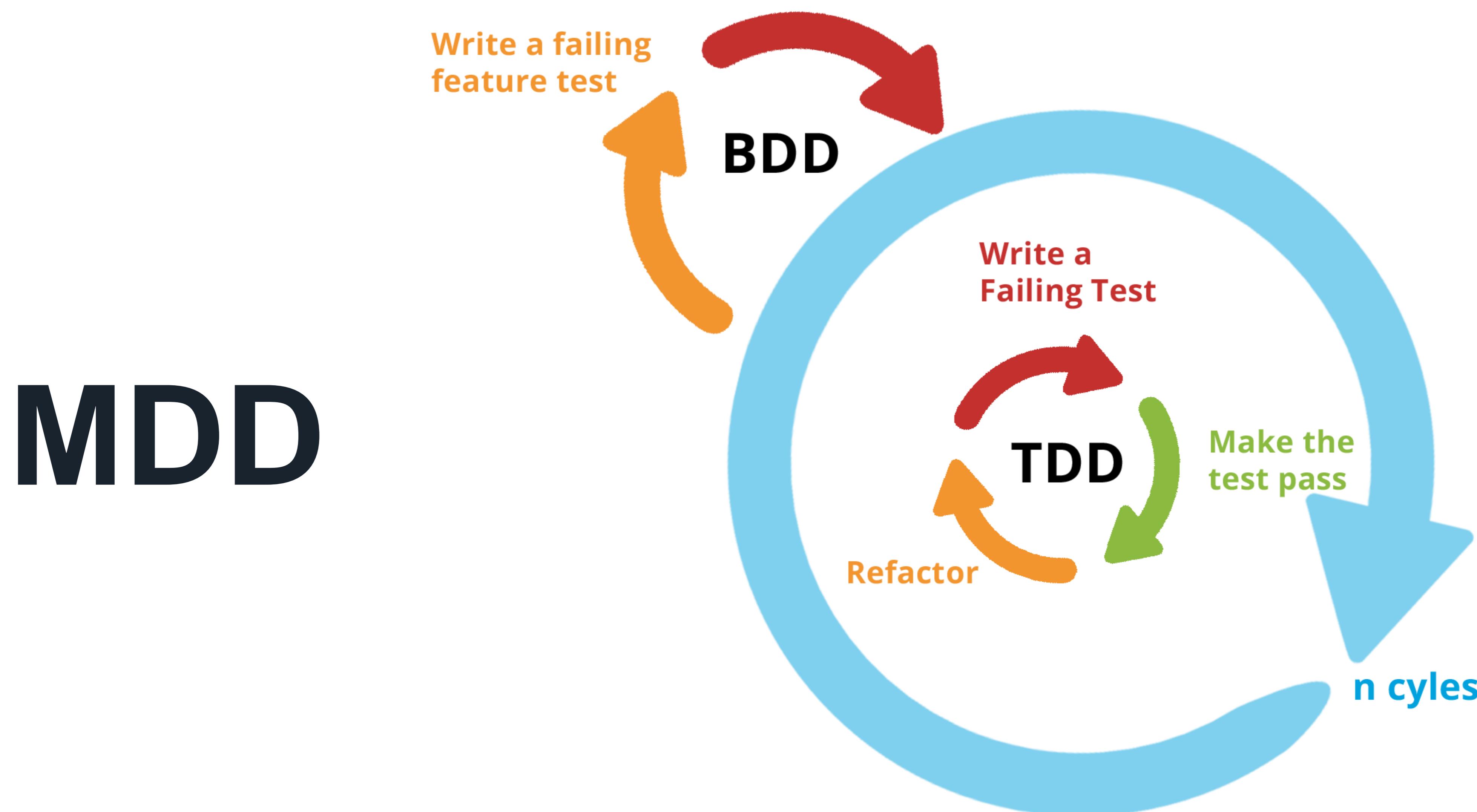
# Парадигмы тестирования



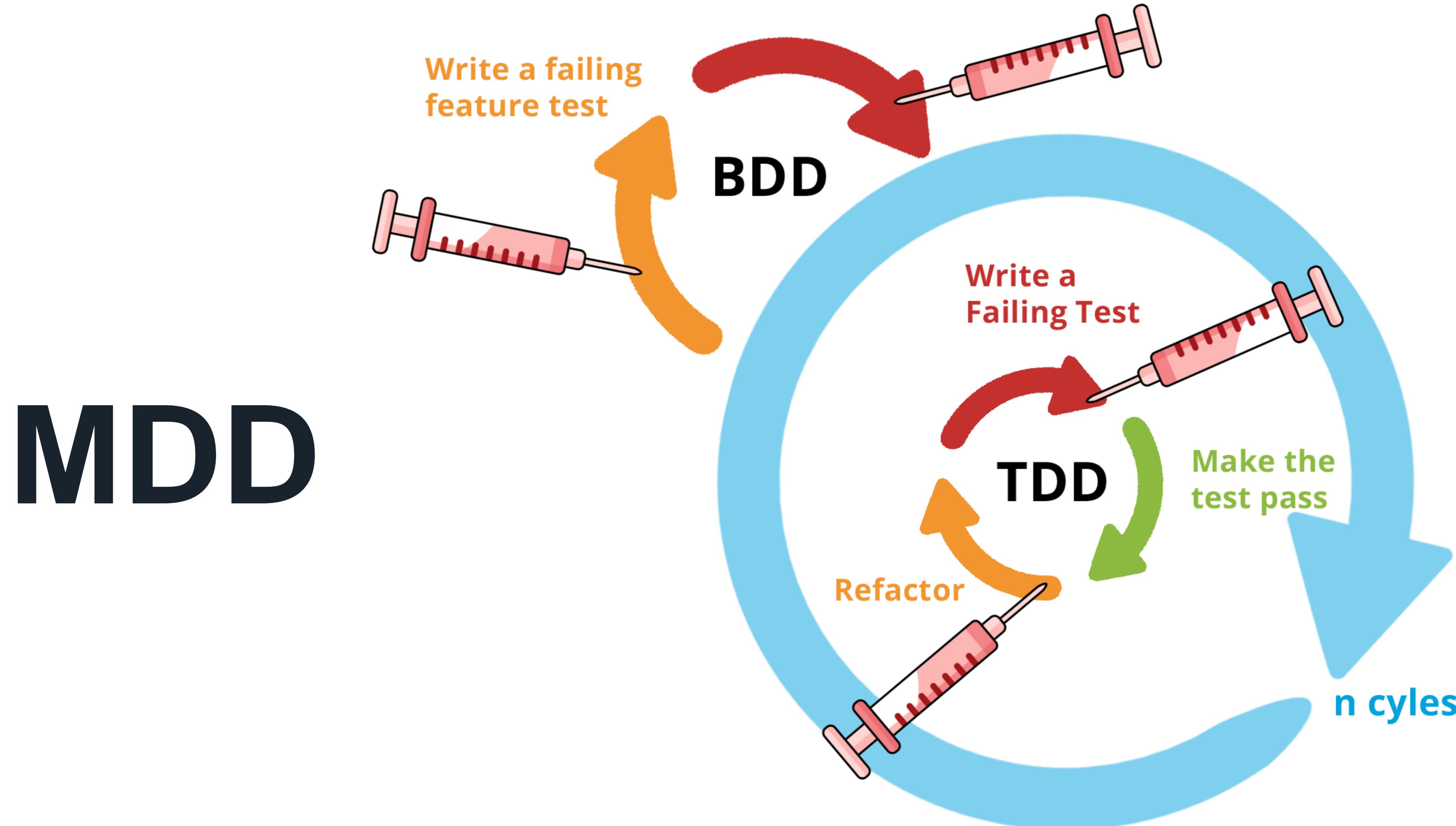
# Парадигмы тестирования



# Парадигмы тестирования

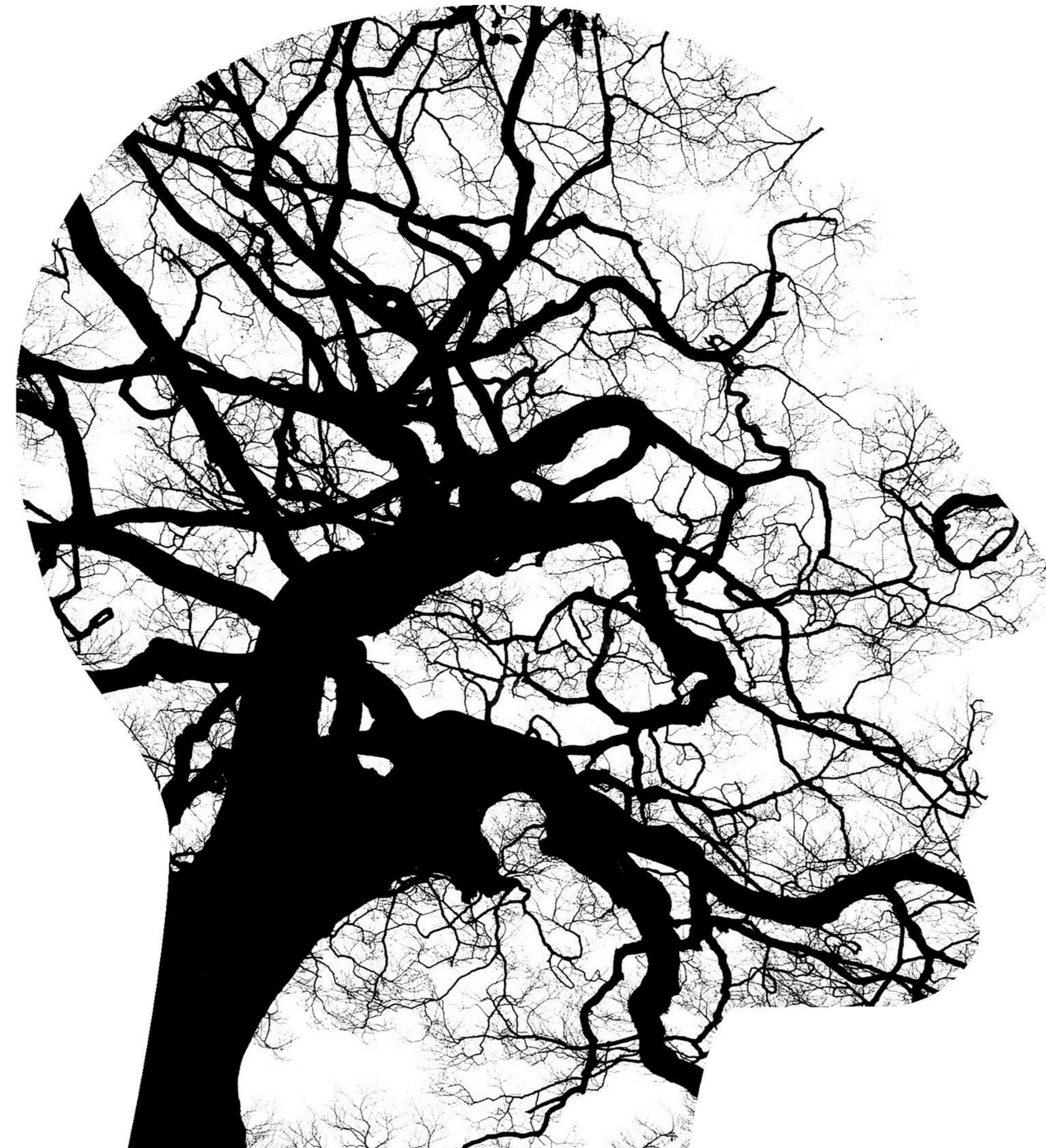


# Парадигмы тестирования



**” MDD (major depressive disorder) – основное  
депрессивное расстройство личности ”**

— Mental and behavioral disorders (F00–F99 & 290–319)



**” Мутационное тестирование - это преднамеренное внесение в код специальных изменений - мутаций с целью проверки программного обеспечения ”**

# Сопротивление

**2 способа**

# Сопротивление

---

## 2 способа

- система типов

# Сопротивление

---

## 2 способа

- система типов
- система тестирования

# JavaScript?



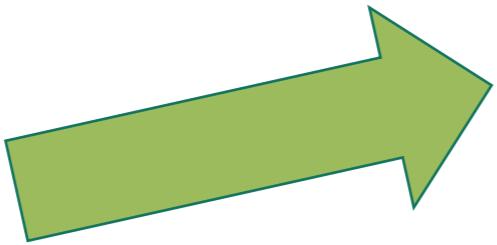
# выживание

---

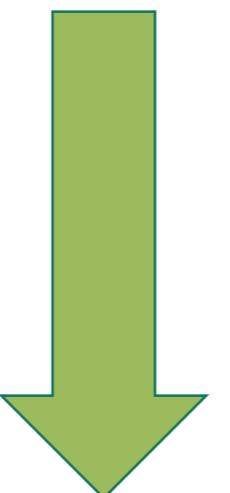


# выживание

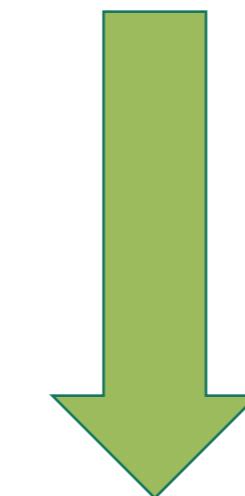
---



# Выживание



# выживание



выживший мутант +1

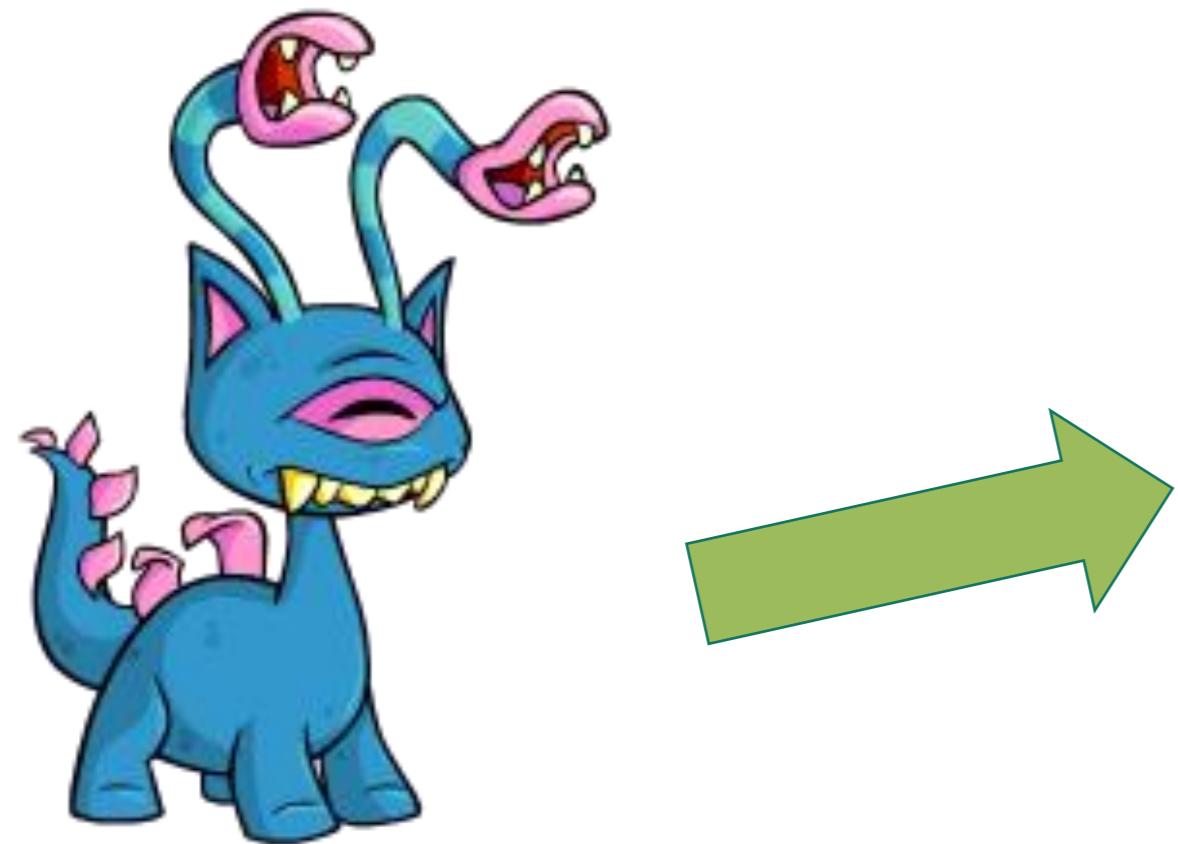
# выживание

---

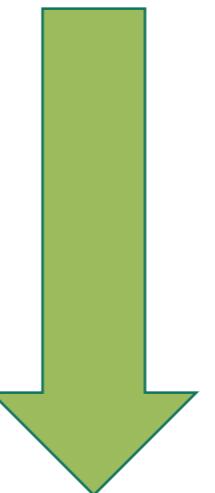


# выживание

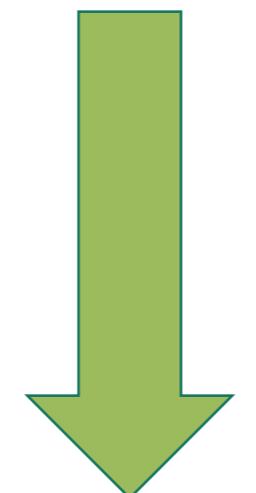
---



# Выживание



# выживание



мертвый мутант +1

# Пример 1

```
int Method(bool a, bool b) {  
    int c;  
    if (a && b) { // mutate to ||  
        c = 1;  
    } else {  
        c = 0;  
    }  
    return c;  
}  
```
```

```
[Fact]  
[InlineData(false, false)]  
void Method_ReturnFalse(bool a, bool b) {  
    // Act  
    var result = Method(a, b);  
  
    // Assert  
    Assert.False(result);  
}
```



# Пример 1

```
int Method(bool a, bool b) {  
    int c;  
    if (a && b) { // mutate to ||  
        c = 1;  
    } else {  
        c = 0;  
    }  
    return c;  
}  
```
```

[Fact]

[InlineData(false, false)]

```
void Method_ReturnFalse(bool a, bool b) {  
    // Act  
    var result = Method(a, b);  
  
    // Assert  
    Assert.False(result);  
}
```



# Пример 1

```
int Method(bool a, bool b) {  
    int c;  
    if (a && b) { // mutate to ||  
        c = 1;  
    } else {  
        c = 0;  
    }  
    return c;  
}  
```
```

```
[Fact]  
[InlineData(false, false)]  
void Method_ReturnFalse(bool a, bool b) {  
    // Act  
    var result = Method(a, b);  
  
    // Assert  
    Assert.False(result);  
}
```



# Пример 1

```
int Method(bool a, bool b) {  
    int c;  
    if (a && b) { // mutate to ||  
        c = 1;  
    } else {  
        c = 0;  
    }  
    return c;  
}  
```
```

```
[Fact]  
[InlineData(false, false)]  
[InlineData(true, false)]  
[InlineData(false, true)]  
void Method_ReturnFalse(bool a, bool b) {  
    // Act  
    var result = Method(a, b);  
  
    // Assert  
    Assert.False(result);  
}
```

3 условия:

1. Покрытие мутанта тестом



# Пример 1

```
int Method(bool a, bool b) {  
    int c;  
    if (a && b) { // mutate to ||  
        c = 1;  
    } else {  
        c = 0;  
    }  
    return c;  
}  
...
```

[Fact]

```
[InlineData(false, false)]  
[InlineData(true, false)]  
[InlineData(false, true)]  
void Method_ReturnFalse(bool a, bool b) {  
    // Act  
    var result = Method(a, b);  
  
    // Assert  
    Assert.False(result);  
}
```

3 условия:

1. Покрытие мутанта тестом
2. Полные входные данные



# Пример 1

```
int Method(bool a, bool b) {  
    int c;  
    if (a && b) { // mutate to ||  
        c = 1;  
    } else {  
        c = 0;  
    }  
    return c;  
}..
```

```
[Fact]  
[InlineData(false, false)]  
[InlineData(true, false)]  
[InlineData(false, true)]  
void Method_ReturnFalse(bool a, bool b) {  
    // Act  
    var result = Method(a, b);  
  
    // Assert  
    Assert.False(result);  
}
```

## 3 условия:

1. Покрытие мутанта тестом
2. Полные входные данные
3. Выходные параметры проверены



# Пример 1

```
int Method(bool a, bool b) {
    int c;
    if (a && b) { // mutate to ||
        c = 1;
    } else {
        c = 0;
    }
    return c;
}..
```

## 3 условия:

1. Покрытие мутанта тестом
2. Полные входные данные
3. Выходные параметры проверены

```
[Fact]
[InlineData(false, false)]
[InlineData(true, false)]
[InlineData(false, true)]
void Method_ReturnFalse(bool a, bool b) {
    // Act
    var result = Method(a, b);

    // Assert
    Assert.False(result);
}
```



# виды мутаций

# Виды мутаций

---

ОПП мутации

# Виды мутаций

---

## ОПП мутации

### 1. Мутации модификаторов доступа

# Виды мутаций

---

## ОПП мутации

1. Мутации модификаторов доступа
2. Мутации наследования

# Виды мутаций

---

## ОПП мутации

1. Мутации модификаторов доступа
2. Мутации наследования
3. Полиморфные мутации

# Виды мутаций

---

## ОПП мутации

1. Мутации модификаторов доступа
2. Мутации наследования
3. Полиморфные мутации
4. Мутации перегрузки методов и их аргументов

# Виды мутаций

---

## ОПП мутации

1. Мутации модификаторов доступа
2. Мутации наследования
3. Полиморфные мутации
4. Мутации перегрузки методов и их аргументов
5. Общие мутации

# Виды мутаций

---

Общие мутации

1. Арифметические операции

- \_\_\_\_\_ +

# Виды мутаций

---

## Общие мутации

1. Арифметические операции
2. Мутации сравнения

> \_\_\_\_\_ => > \_\_\_\_\_ <

# Виды мутаций

---

## Общие мутации

1. Арифметические операции
2. Мутации сравнения
3. Мутации логических операторов

&& \_\_\_\_\_ ||

# Виды мутаций

---

## Общие мутации

1. Арифметические операции
2. Мутации сравнения
3. Мутации логических операторов
4. Мутации bool типа

true \_\_\_\_\_ false

# Виды мутаций

## Общие мутации

1. Арифметические операции
2. Мутации сравнения
3. Мутации логических операторов
4. Мутации bool типа
5. Унарные операции

$\text{++i}$  \_\_\_\_\_  $\text{i++}$

$\text{++i}$  \_\_\_\_\_  $\text{--i}$

# Виды мутаций

---

## Общие мутации

1. Арифметические операции
2. Мутации сравнения
3. Мутации логических операторов
4. Мутации bool типа
5. Унарные операции
6. Строковые мутации

“”

“asd”

“asd”

# Виды мутаций

---

## Общие мутации

1. Арифметические операции
2. Мутации сравнения
3. Мутации логических операторов
4. Мутации bool типа
5. Унарные операции
6. Строковые мутации
7. Мутации массивов и списков

[0,1,2] \_\_\_\_\_ [0,0,2]

# ВИДЫ МУТАЦИЙ

---

LINQ мутации

Last \_\_\_\_\_ First

Any \_\_\_\_\_ All

First \_\_\_\_\_ Single

FirstOrDefault \_\_\_\_\_ SingleOrDefault

# Статистика

---

## Flurl

9000 строк

85%

300 тестов

43 сек

363 мутанта

30 минут

2 критических бага

## Ecwid

5000 строк

90%

200 тестов

16 сек

300 мутантов

15 минут

4 критических бага

## статистика выживших мутаций



мертвый код



тесты



эквивалентные

# примеры мутаций

# строковые мутации



## Пример 2

```
private bool ExceptionMethod(int a) {
    if (a < 0)

    ...

    throw new ArgumentException("USER_DB is not path pattern uiier_ksks",
    "user-name");

    ...

    return true;
}

public string LibraryMethod(int a) {
    try {
        if (ExceptionMethod(a)) return "Successfully";
    }
    catch (ArgumentException e) {
        throw new Exception("Something happened", e);
    }
    return null;
}
```



## Пример 2

```
private bool ExceptionMethod(int a) {
    if (a < 0)

    ...

    throw new ArgumentException("USER_DB is not path pattern uiier_ksks",
"user-name");

    ...

    return true;
}

public string LibraryMethod(int a) {
    try {
        if (ExceptionMethod(a)) return "Successfully";
    }
    catch (ArgumentException e) {
        throw new Exception("Something happened", e);
    }
    return null;
}
```



## Пример 2

---

[Fact]

```
public void LibraryMethod_NegativeData_ThrowException() {
    // Act
    var ex = Assert.Throws<Exception>(() => SampleClass.LibraryMethod(-1));
    // Assert
    Assert.Equal("Something happened", ex.Message);
}
```



## Пример 2

[Fact]

```
public void LibraryMethod_NegativeData_ThrowException() {
    // Act
    var ex = Assert.Throws<Exception>(() => SampleClass.LibraryMethod(-1));
    // Assert
    Assert.Equal("Something happened", ex.Message);
}
```

Что забыли?



## Пример 2

```
private bool ExceptionMethod(int a) {
    if (a < 0)

    ...

    throw new ArgumentException("USER_DB is not path pattern uiier_ksks",
    "user-name");

    ...

    return true;
}

public string LibraryMethod(int a) {
    try {
        if (ExceptionMethod(a)) return "Successfully";
    }
    catch (ArgumentException e) {
        throw new Exception("Something happened", e);
    }
    return null;
}
```



## Пример 2

```
private bool ExceptionMethod(int a) {
    if (a < 0)
    ...
    throw new ArgumentException("USER_DB is not path pattern uiier_ksks",
"user-name");
    ...
    return true;
}

public string LibraryMethod(int a) {
    try {
        if (ExceptionMethod(a)) return "Successfully";
    }
    catch (ArgumentException e) {
        throw new Exception("Something happened", e);
    }
    return null;
}
```

МУТИРУЕТ В “”



## Пример 2

```
[Fact]
public void LibraryMethod_NegativeData_ThrowException() {
    // Act
    var ex = Assert.Throws<Exception>(() => SampleClass.LibraryMethod(-1));
    // Assert

    Assert.Equal("Something happened", ex.Message);
    Assert.Equal("USER_DB is not path pattern uiier_ksks\nParameter name:
    user-name", ex.InnerException.Message);
}
```



## Пример 3

[Fact]

```
public void Credentials_Fail() {  
    Assert.Throws<Exception>(() => new Ecwid(0, Token, Token));  
  
    Assert.Throws<Exception>(() => new Ecwid(ShopId));  
  
    Assert.Throws<Exception>(() => new Ecwid(ShopId, " ", Token));  
  
    Assert.Throws<Exception>(() => new Ecwid(ShopId, Token, " "));  
}
```



# мутации сравнения



## Пример 4

```
public string Encode(string s, ...) {  
    if (string.IsNullOrEmpty(s)) return s;  
  
    if (s.Length > MAX_URL_LENGTH) {  
        ...  
    }  
    ...  
}
```



## Пример 5

```
public| Builder Limit(this Builder query, int limit) {  
    if (limit <= 0)  
        throw new ArgumentException("Limit must be greater than 0",  
            nameof(limit));  
  
    if (limit > 100) limit = 100;  
  
    query.AddOrUpdate("limit", limit);  
    return query;  
}
```

Входные данные для теста были **[-1, 101]**



## Пример 5

```
public Builder Limit(this Builder query, int limit) {  
    if (limit <= 0)                                limit < 0  
        throw new ArgumentException("Limit must be greater than 0",  
            nameof(limit));  
  
    if (limit > 100) limit = 100;                  limit < 100  
  
    query.AddOrUpdate("limit", limit);  
    return query;  
}
```

Входные данные для теста были **[-1, 101]**



## Пример 5

```
public Builder Limit(this Builder query, int limit) {  
    if (limit <= 0)                                limit < 0  
        throw new ArgumentException("Limit must be greater than 0",  
            nameof(limit));  
  
    if (limit > 100) limit = 100;                  limit < 100  
  
    query.AddOrUpdate("limit", limit);  
    return query;  
}
```

Входные данные для теста были **[-1, 101] + 0 + 12**



## Пример 5

```
public| Builder Limit(this Builder query, int limit) {  
    if (limit <= 0)  
        throw new ArgumentException("Limit must be greater than 0",  
            nameof(limit));  
  
    if (limit > 100) limit = 100;  
  
    query.AddOrUpdate("limit", limit);  
    return query;  
}
```

Входные данные для теста были [-1, 0, 12, 101]



## Пример 5

```
public Builder Limit(this Builder query, int limit) {  
    if (limit <= 0)  
        throw new ArgumentException("Limit must be greater than 0",  
            nameof(limit));  
  
    if (limit > 100) limit = 100;      limit >= 100  
  
    query.AddOrUpdate("limit", limit);  
    return query;  
}
```

Входные данные для теста были **[-1, 0, 12, 101]**



## Пример 5

```
public Builder Limit(this Builder query, int limit) {  
    if (limit <= 0)  
        throw new ArgumentException("Limit must be greater than 0",  
            nameof(limit));  
  
    if (limit > 100) limit = 100;      limit >= 100  
  
    query.AddOrUpdate("limit", limit);  
    return query;  
}
```

Входные данные для теста были **[-1, 0, 12, 101] + 99**



## Пример 5

```
public Builder Limit(this Builder query, int limit) {  
    if (limit <= 0)  
        throw new ArgumentException("Limit must be greater than 0",  
            nameof(limit));  
  
    if (limit > 99) limit = 100;  
  
    query.AddOrUpdate("limit", limit);  
    return query;  
}
```

Входные данные для теста были **[-1, 0, 12, 99, 101]**



## Пример 6

```
public void Some<T>(object arg, IEnumerable<T> list) {  
    ...  
    var out = new List<T>();  
    ...  
  
    if (out.Count > 0) {  
        ...  
    }  
    ...  
}
```



## Пример 6

```
public void Some<T>(object arg, IEnumerable<T> list) {  
    ...  
    var out = new List<T>();  
    ...  
  
    if (out.Count > 0) {    >= 0  
        ...  
    }  
    ...  
}
```



## Пример 6

```
public void Some<T>(object arg, IEnumerable<T> list) {  
    ...  
    var out = new List<T>();  
    ...  
  
    if (out.Any()) { // FirstOrDefault() != null  
        ...  
    }  
    ...  
}
```



## Пример 7

```
public T GetValue(int field) {  
    var values = await GetValues(new {field});  
  
    return values.FirstOrDefault();  
}
```

[Fact]

```
public void GetValue_ReturnCorrectData() {  
    //... mock data  
    var result = await client.GetValue("some data");  
  
    Assert.Equal("some data", result.ValueNumber);  
    //... some other assertion  
}
```



## Пример 7

```
public T GetValue(int field) {  
    var values = await GetValues(new {field});  
  
    return values.FirstOrDefault();  
}
```

[Fact]

```
public void GetValue_ReturnCorrectData() {  
    //... mock data  
    var result = await client.GetValue("some data");  
  
    Assert.Equal("some data", result.ValueNumber);  
    //... some other assertion  
}
```



## Пример 7

```
public T GetValue(int field) {  
    var values = await GetValues(new {field});  
  
    return values.FirstOrDefault();      LastOrDefault()  
}  
  
[Fact]  
public void GetValue_ReturnCorrectData() {  
    //... mock data  
    var result = await client.GetValue("some data");  
  
    Assert.Equal("some data", result.ValueNumber);  
    //... some other assertion  
}
```



## Пример 7

```
public T GetValue(int field) {  
    var values = await GetValues(new {field});  
  
    return values.FirstOrDefault();  
}  
  
[Fact]  
public void GetValue_ReturnCorrectData() {  
    //... mock data  
    var result = await client.GetValue("some data");  
  
    Assert.Equal("some data", result.ValueNumber);  
    //... some other assertion  
}
```



# **эквивалентная мутация**

## Пример 8

---

```
int index = 0;  
while (...) {  
    // ...;  
    index++;  
    if (index == 10) { // mutate to >=  
        break;  
    }  
}
```



# Пример 9

```
public void Remove(SomeType<T> container, T data) {  
    if (data == null) return;  
  
    if (container.All(c => c.SomeField != data.SomeField)) return;  
  
    container.Values.Remove(data); // очень много связанный логики и т.п.  
}  
  
[Fact]  
public void Remove_NotExist_NotRemoved() {  
    // Arrange  
    var expected = container.Values.Count;  
    var data = new Value();  
  
    // Act  
    TRepository.Remove(container, data);  
  
    // Assert  
    Assert.Equal(expected, container.Values.Count);  
}
```



# Пример 9

```
public void Remove(SomeType<T> container, T data) {  
    if (data == null) return;  
  
    if (container.All(c => c.SomeField != data.SomeField)) return; мутит в ==  
  
    container.Values.Remove(data); // очень много связанный логики и т.п.  
}  
  
[Fact]  
public void Remove_NotExist_NotRemoved() {  
    // Arrange  
    var expected = container.Values.Count;  
    var data = new Value();  
  
    // Act  
    TRepository.Remove(container, data);  
  
    // Assert  
    Assert.Equal(expected, container.Values.Count);  
}
```



# Пример 9

```
public void Remove(SomeType<T> container, T data) {  
    if (data == null) return;  
  
    var deleteT = container.FirstOrDefault(c => c.SomeField ==  
        data.SomeField);  
    if (deleteT == null) return;  
  
    container.Values.Remove(deleteT); // очень много связанной логики и т.п.  
}
```

[Fact]

```
public void Remove_Exist_Removed() {  
    // Arrange  
    var expected = container.Values.Count;  
    var data = container.Values.Get(1);  
  
    // Act  
    TRepository.Remove(container, data);  
  
    // Assert  
    Assert.Equal(expected - 1, container.Values.Count);  
}
```



## Пример 9

```
public void Remove(SomeType<T> container, T data) {  
    if (data == null) return;  
  
    var deleteT = container.FirstOrDefault(c => c.SomeField ==  
        data.SomeField);  
    if (deleteT == null) return;  
  
    container.Values.Remove(deleteT); // очень много связанной логики и т.п.  
}
```

[Fact]

```
public void Remove_Exist_Removed() {  
    // Arrange  
    var expected = container.Values.Count;  
    var data = container.Values.Get(1);  
  
    // Act  
    TRepository.Remove(container, data);  
  
    // Assert  
    Assert.Equal(expected - 1, container.Values.Count);  
}
```



## Пример 9

```
public void Remove(SomeType<T> container, T data) {  
    if (data == null) return;  
  
    var deleteT = container.FirstOrDefault(c => c.SomeField ==  
        data.SomeField);  
    if (deleteT == null) return;  
  
    container.Values.Remove(deleteT); // очень много связанной логики и т.п.  
}
```

[Fact]

```
public void Remove_Exist_Removed() {  
    // Arrange  
    var expected = container.Values.Count;  
    var someField = container.Values.Get(1).SomeField;  
    var data = new Value(someField);  
  
    // Act  
    TRepository.Remove(container, data);  
  
    // Assert  
    Assert.Equal(expected - 1, container.Values.Count);  
}
```



## Пример 9

```
public void Remove(SomeType<T> container, T data) {  
    if (data == null) return;  
  
    var deleteT = container.FirstOrDefault(c => c.SomeField ==  
        data.SomeField);  
    if (deleteT == null) return;  
  
    container.Values.Remove(deleteT); // очень много связанной логики и т.п.  
}
```

мутирует в SingleOrDefault

```
[Fact]  
public void Remove_Exist_Removed() {  
    // Arrange  
    var expected = container.Values.Count;  
    var someField = container.Values.Get(1).SomeField;  
    var data = new Value(someField);  
  
    // Act  
    TRepository.Remove(container, data);  
  
    // Assert  
    Assert.Equal(expected - 1, container.Values.Count);  
}
```



# Пример 9

```
public void Remove(SomeType<T> container, T data) {  
    if (data == null) return;  
  
    var deleteT = container.SingleOrDefault(c => c.SomeField ==  
        data.SomeField);  
    if (deleteT == null) return;  
  
    container.Values.Remove(deleteT); // очень много связанной логики и т.п.  
}
```

[Fact]

```
public void Remove_ExistDouble_ThrowsException() {  
    // Arrange  
    var expected = container.Values.Count;  
    var someField = container.Values.Get(1).SomeField;  
    var data = new Value(someField);
```

```
// Act  
Assert.Throws<InvalidOperationException>(() =>  
    _TRepository.Remove(container, data));
```

```
// Assert  
Assert.Equal(expected, container.Values.Count);  
}
```



# эквивалентная мутация

>>

рефакторинг



Powered by  
**InfoSupport**

<https://Stryker-mutator.io/>



Powered by  
**InfoSupport**

# stryker

## Особенности

1. Покрыт тестами, в том числе мутационными

# stryker

---

## Особенности

1. Покрыт тестами, в том числе мутационными
2. Работает сейчас только для .NET Core

# stryker

---

## Особенности

1. Покрыт тестами, в том числе мутационными
2. Работает сейчас только для .NET Core
3. Только общие мутации и LINQ мутации

# stryker

---

## Особенности

1. Покрыт тестами, в том числе мутационными
2. Работает сейчас только для .NET Core
3. Только общие мутации и LINQ мутации
4. Поддержка конфигурации

# stryker

---

## Особенности

1. Покрыт тестами, в том числе мутационными
2. Работает сейчас только для .NET Core
3. Только общие мутации и LINQ мутации
4. Поддержка конфигурации
5. Отключение части мутаторов

# stryker

---

## Особенности

1. Покрыт тестами, в том числе мутационными
2. Работает сейчас только для .NET Core
3. Только общие мутации и LINQ мутации
4. Поддержка конфигурации
5. Отключение части мутаторов
6. Ограничение на проекты или на файлы

# stryker

---

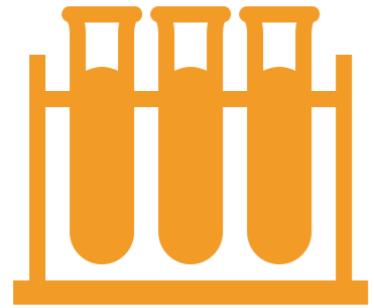
## Особенности

1. Покрыт тестами, в том числе мутационными
2. Работает сейчас только для .NET Core
3. Только общие мутации и LINQ мутации
4. Поддержка конфигурации
5. Отключение части мутаторов
6. Ограничение на проекты или на файлы
7. Флаг красного билда

# stryker

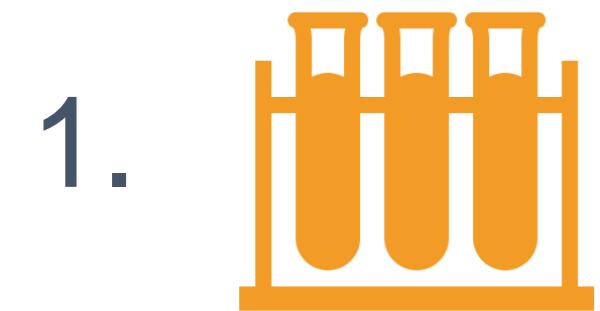
---

1.



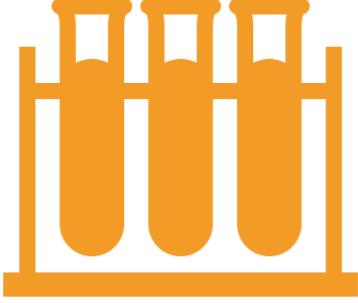
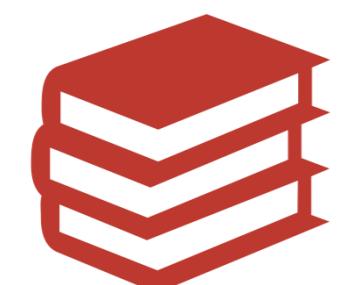
# stryker

---



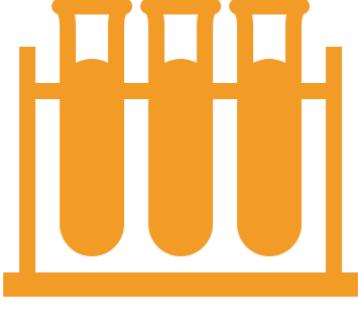
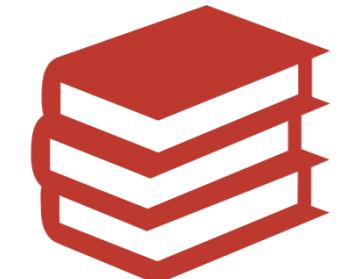
# stryker

---

1. 
2. 
3.  *Microsoft.CodeAnalysis.Csharp.***SharpSyntaxTree**

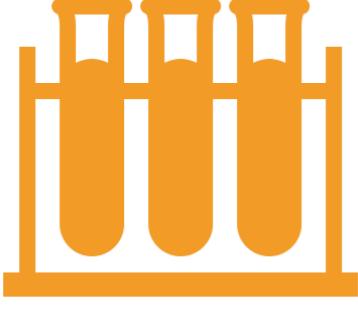
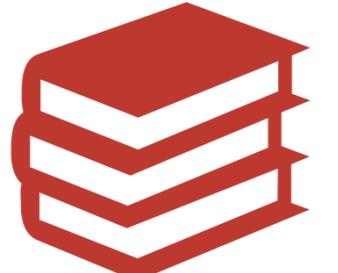
# stryker

---

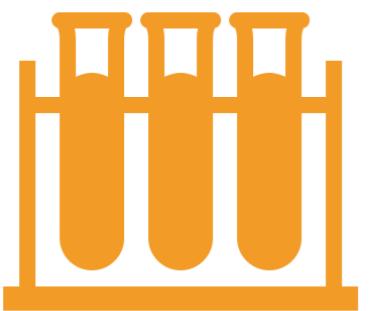
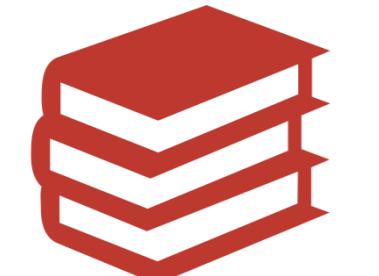
1. 
2. 
3.  *Microsoft.CodeAnalysis.Csharp.***SharpSyntaxTree**
4.   
A large teal arrow points from the icon in item 1 to the icon in item 4.

# stryker

---

1. 
2. 
3. 
4. 
5.  *Mutate recursion*

# stryker

1. 
2. 
3. 
4. 
5. 
6.   
*CSharpCompilation*

# stryker



4.



5.



6.



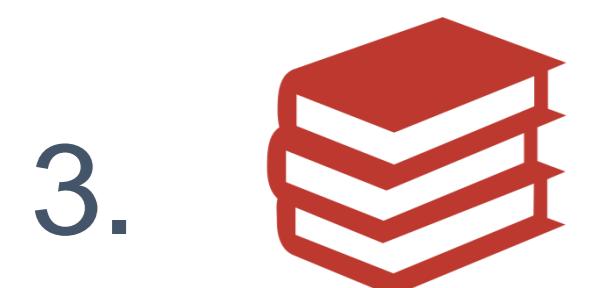
7.



Инъекция в код

# stryker

---



4.



5.



6.



7.



8.



TPL n-threads

# stryker



4.



5.



6.



7.



8.



9.



Долгожданная награда

demo

# критерии применимости

## критерии применимости

1. Покрытие кода тестами  $>70\%$

## критерии применимости

1. Покрытие кода тестами  $>70\%$
2. Скорость работы тестов

## критерии применимости

---

1. Покрытие кода тестами **>70%**
2. Скорость работы тестов
3. Чистые маленькие методы

## кriterии применимости

---

1. Покрытие кода тестами **>70%**
2. Скорость работы тестов
3. Чистые маленькие методы
4. .NET Core

## критерии применимости

---

1. Покрытие кода тестами **>70%**
2. Скорость работы тестов
3. Чистые маленькие методы
4. .NET Core
5. Сила духа и авантюризм

# Pipeline?

# Выводы

# Выводы

---

MDD  
классно подходит при работе  
с парадигмой **TDD**  
при написании **новых** тестов  
и  
**однократного**  
аудита текущей системы

# Выводы

---

для поиска  
**неочевидных**  
проблемных мест  
и  
**оптимизации** юнит тестов.

# Выводы

---

MDD подходит  
для  
аудита кода  
внешних разработчиков

# Выводы

---

MDD  
подходит  
для  
проверки себя  
и  
внутренних разработчиков

# Выводы

---

MDD  
подходит  
для  
погружения новичков  
в проект

**КТО ГОТОВ ПРИМЕНИТЬ?**

