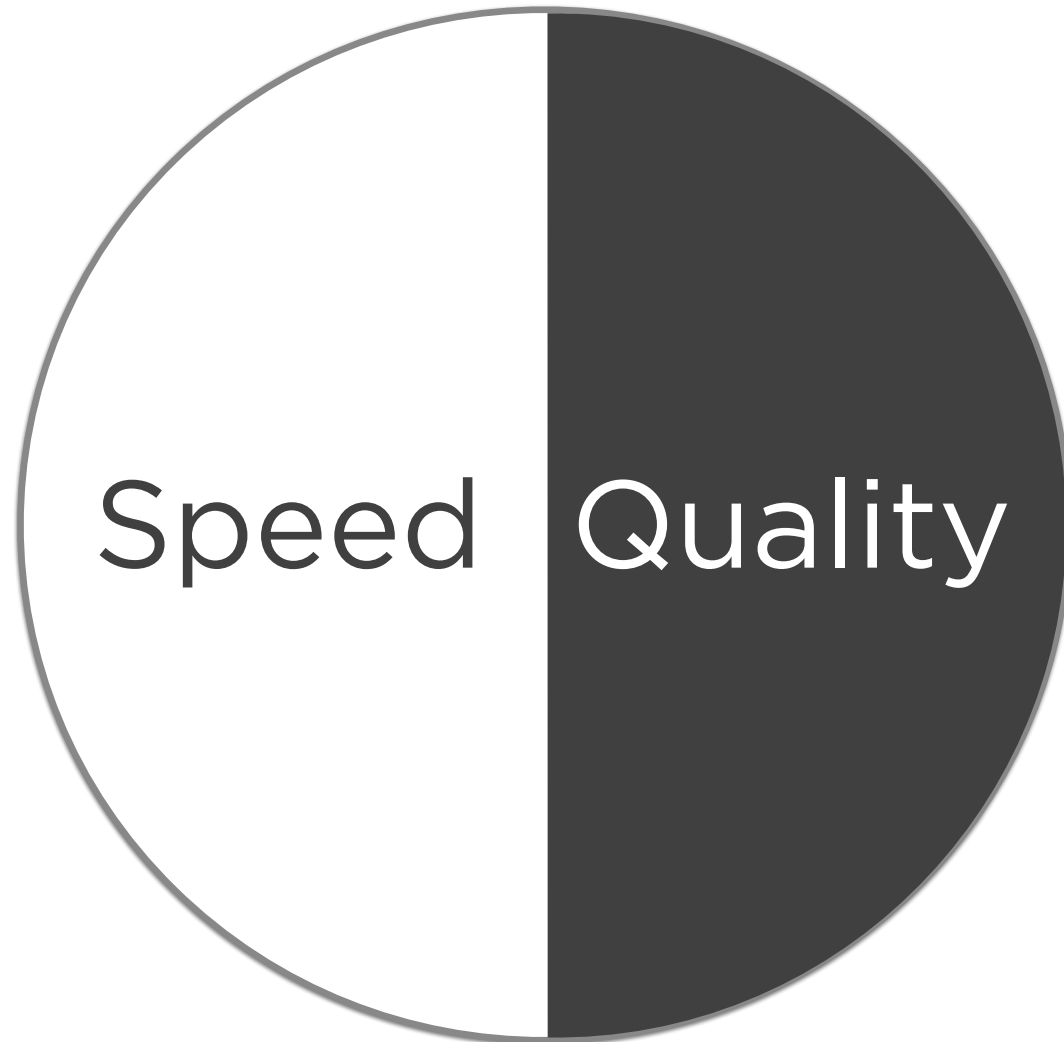
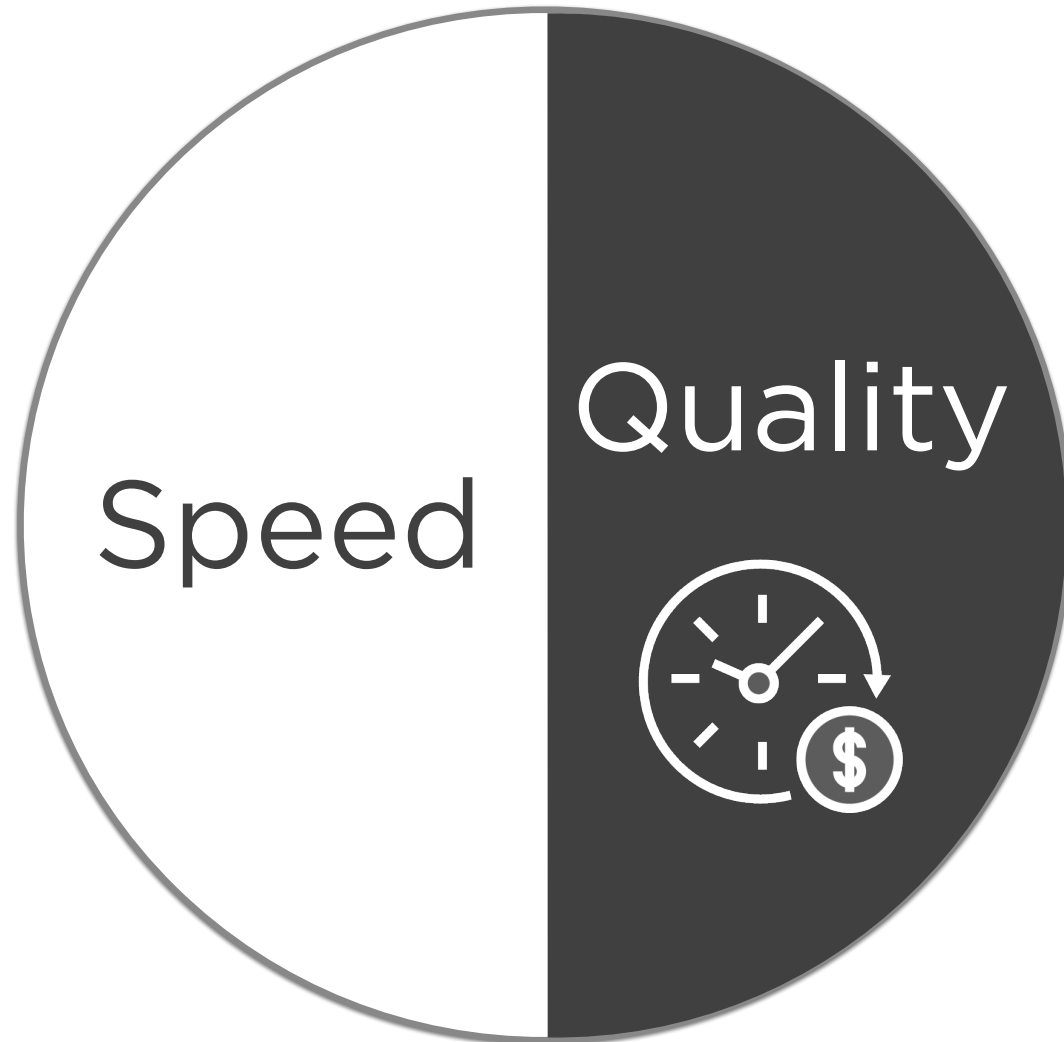
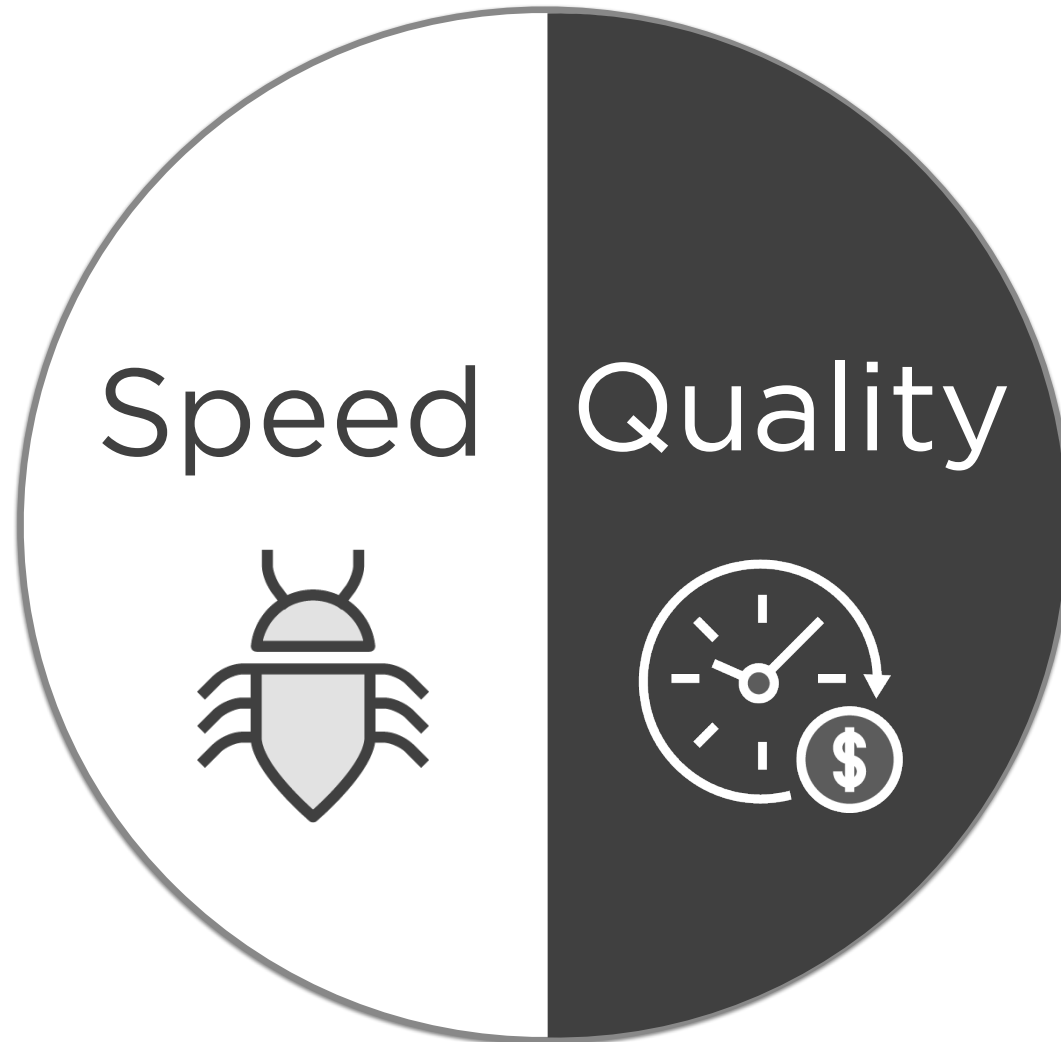


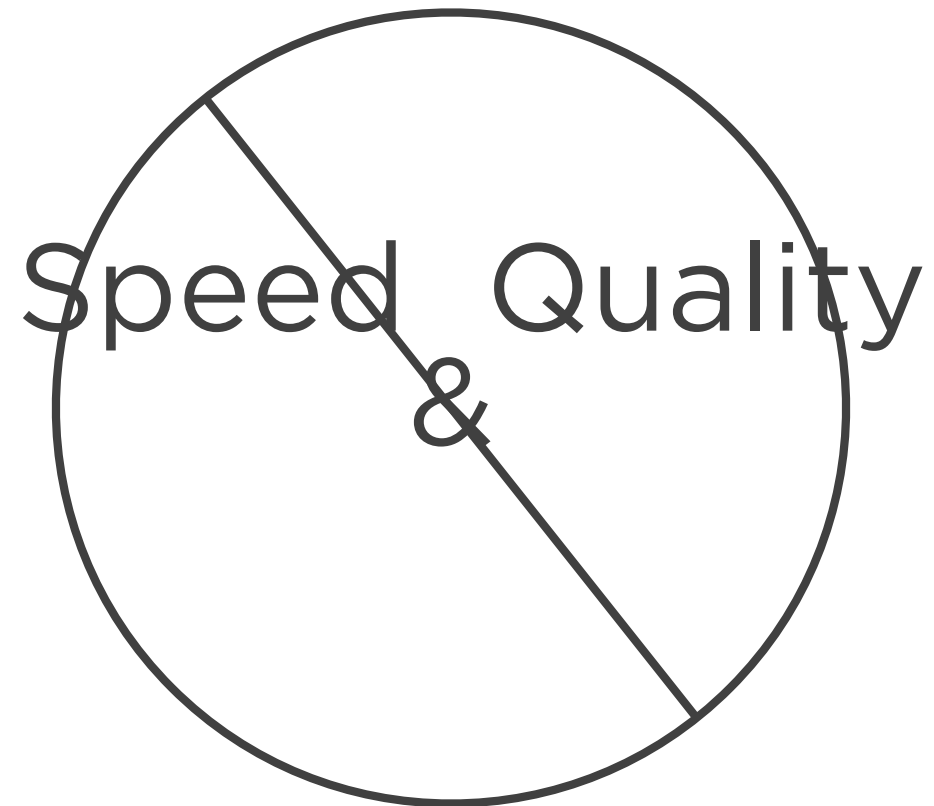
What Is TDD and Why It Is Not Unit Testing: Executive Briefing

WHAT TDD IS AND WHAT IT IS NOT









“The only way to go **fast**
is to go **well.**”

Robert Martin

Are The Claims True?



Defects



Productivity



Quality



Test Driven Development



Test Driven Development

Definitions

- ☐ Automated Tests
- ☐ Unit Testing
- ☐ Test-last
- ☐ Test-first
- ☐ Code Coverage

I test code.

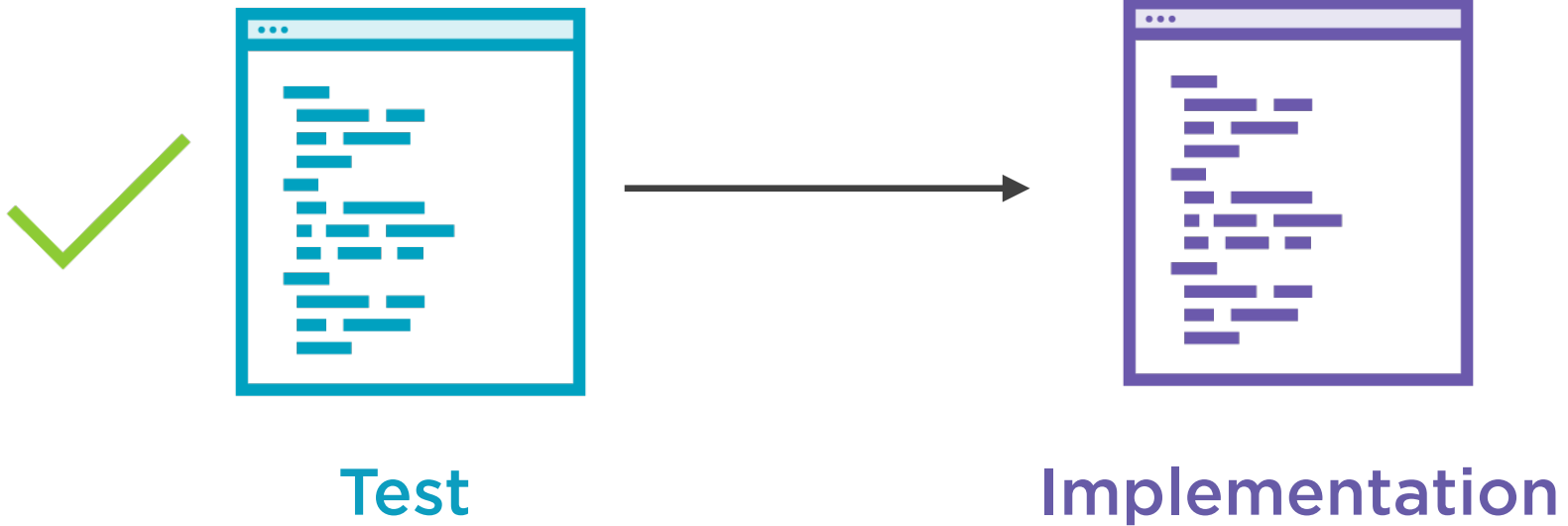


I have tests.



I have tests.
They're automatically
repeated.

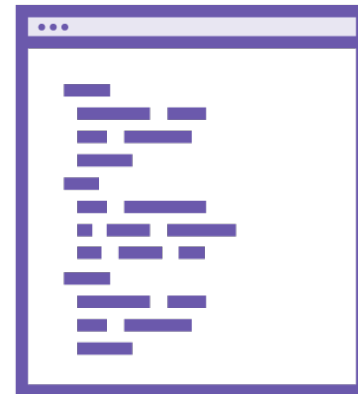
Automated Testing



Automated Testing



Test

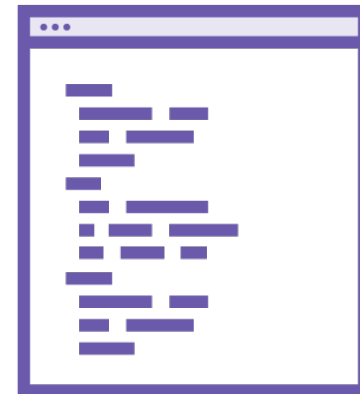


Implementation

Automated Testing



Test

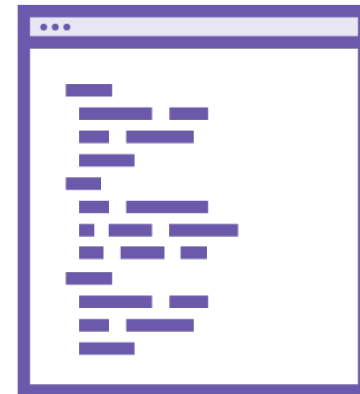


Implementation

Automated Testing



**Test
Suite**

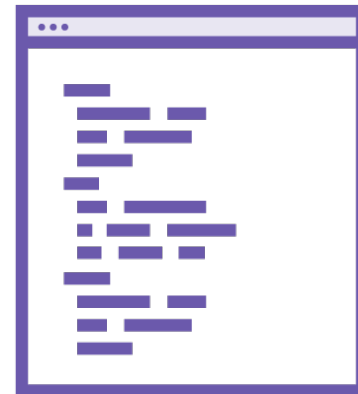


Implementation

Automated Testing



Customer

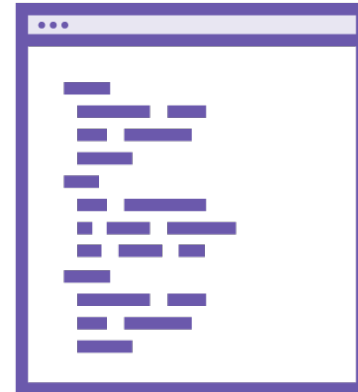


Implementation

Automated Testing



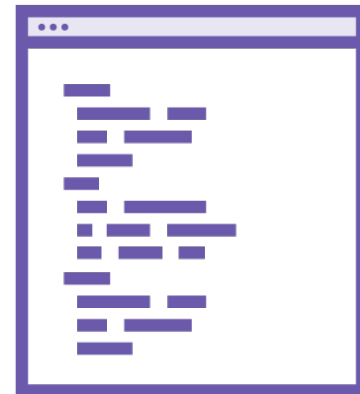
Customer



Automated Testing



Customer



Production Code

Definitions

- ☐ Automated Tests
- ☐ Unit Testing
- ☐ Test-last
- ☐ Test-first
- ☐ Code Coverage

Definitions

☒ **Automated Tests**

☐ **Unit Testing**

☐ **Test-last**

☐ **Test-first**

☐ **Code Coverage**

The devs broke
scenario A.... again

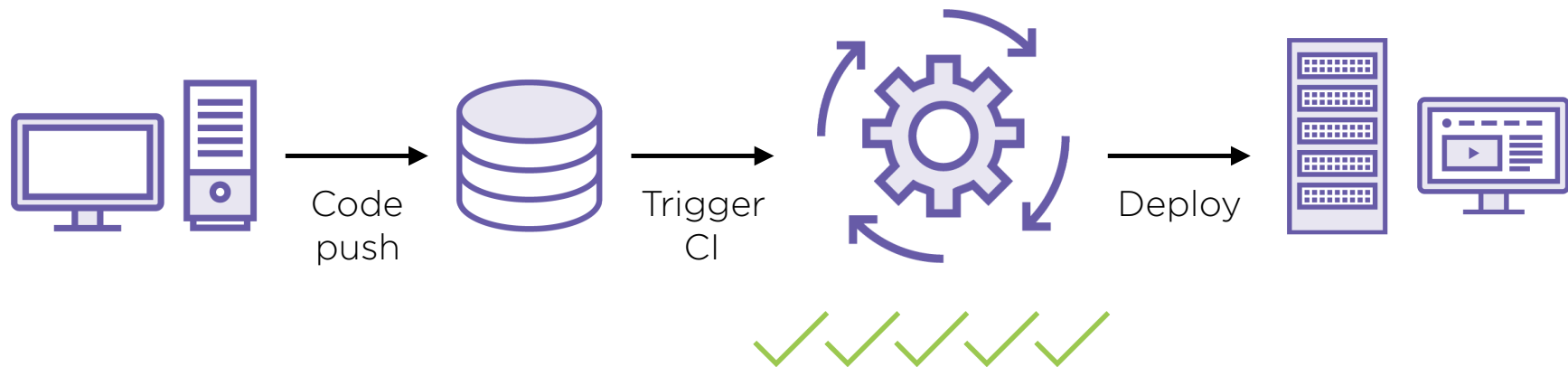


I wonder what
happens when I...
start task A and B at
the same time



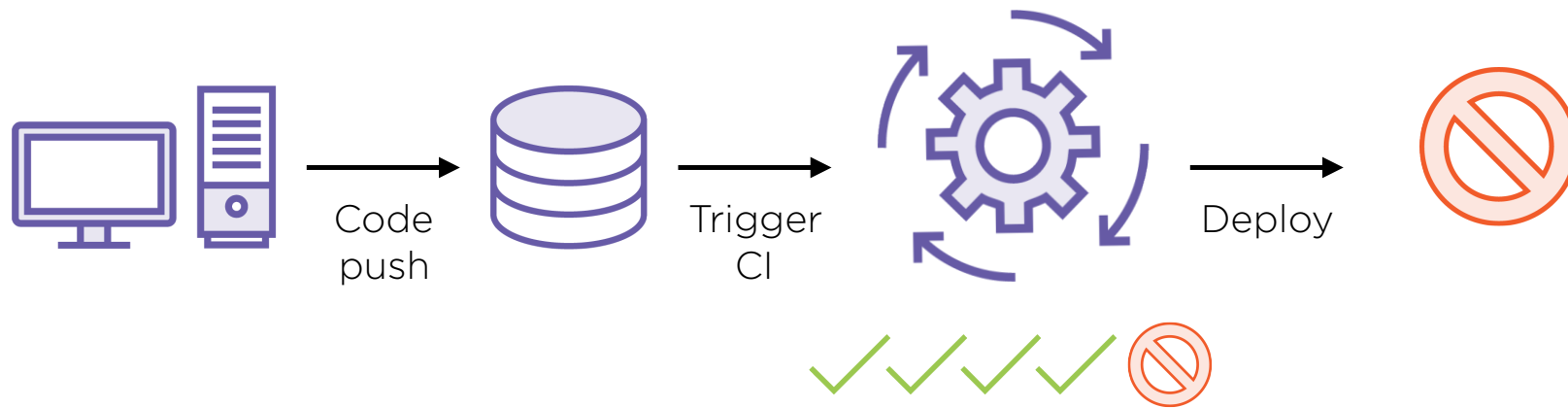
Continuous Integration

This gear icon is intentionally large. I want the emphasis to be on the CI part. Is that okay? I don't feel very strongly about it.



Continuous Integration

This gear icon is intentionally large. I want the emphasis to be on the CI part. Is that okay? I don't feel very strongly about it.







No time to test.
Just ship it!

Red, Green, Refactor



bedtime routine

1. Choose a small task

Tests

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
}
```

2. Write a failing test

Tests

...

...

```
new_test = function()  
{  
  routine_includes_brushing_teeth();  
}
```

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
}
```


Tests



...



...

 `new_test = function()
{

 routine_includes_brushing_teeth();

}`

Implementation

```
bedtime_routine = function() {  
  
    eat_snack();  
    put_on_pajamas();  
  
}
```

2. Write a failing test

2. Write a failing (RED) test

3. Write simplest code to make the test pass


Tests



...



...

 `new_test = function()
{
 routine_includes_brushing_teeth();
}`

Implementation

```
bedtime_routine = function() {  
    eat_snack();  
    put_on_pajamas();  
    brush_teeth();  
}
```

Tests

✓ ...
✓ ...

✓ new_test = function()
 {
 routine_includes_brushing_teeth();

 }

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
  brush_teeth();  
}
```

3. Write simplest code to
make the test pass (GREEN)

4. Refactor

Tests

✓ ...
✓ ...

✓ `new_test = function()
{
 routine_includes_brushing_teeth();

}`

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
  brush_teeth();  
}
```

Tests

✓ ...
✓ ...

✓ `new_test = function()
{
 routine_includes_brushing_teeth();
}`

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
  brush_  
}
```

Tests

✓ ...
✓ ...

✓ `new_test = function()
{
 routine_includes_brushing_teeth();
}`

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
  brush_all_teeth();  
}
```

1. Choose a small task
2. Write a failing (RED) test
3. Write simplest code to make the test pass (GREEN)
4. Refactor
5. Repeat

Tests



...



...


```
✓ new_test = function()  
{  
  routine_includes_brushing_teeth();  
}
```

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
  brush_all_teeth();  
}
```

Tests




new_test = function()
{
 routine_includes_getting_in_bed();
}

Implementation

```
bedtime_routine = function() {  
    eat_snack();  
    put_on_pajamas();  
    brush_all_teeth();  
}
```

Tests



new_test = function()
{
 routine_includes_getting_in_bed();
}

Implementation

```
bedtime_routine = function() {  
    eat_snack();  
    put_on_pajamas();  
    brush_all_teeth();  
    get_in_bed();  
}
```


Tests

✓ ...
✓ ...
✓ ...

```
✓ new_test = function()  
{  
  routine_includes_getting_in_bed();  
}
```

Implementation

```
bedtime_routine = function() {  
  eat_snack();  
  put_on_pajamas();  
  brush_all_teeth();  
  get_in_bed();  
}
```

1. Choose a small task
2. Write a failing (RED) test
3. Write simplest code to make the test pass (GREEN)
4. Refactor
5. Repeat

1. Choose a small task
2. Write a failing **(RED)** test
3. Write simplest code to make the test pass (GREEN)
4. Refactor
5. Repeat

1. Choose a small task
2. Write a failing **(RED)** test
3. Write simplest code to make the test pass **(GREEN)**
4. Refactor
5. Repeat

1. Choose a small task
2. Write a failing **(RED)** test
3. Write simplest code to make the test pass **(GREEN)**
4. **Refactor**
5. Repeat

Definitions

- ☒ Automated Tests
- ☐ Unit Testing
- ☐ Test-last
- ☐ Test-first
- ☐ Code Coverage

Definitions

- ☒ Automated Tests
- ☒ **Unit Testing**
- ☐ Test-last
- ☐ Test-first
- ☐ Code Coverage

Definitions

- ☒ Automated Tests
- ☒ Unit Testing
- ☐ **Test-last / test-after**
- ☐ Test-first
- ☐ Code Coverage

Tests

test_1...

test_2...

test_3...

test_4...

Implementation

```
foo = function() {  
  ...  
}
```

```
bar = function() {  
  ...  
}
```

Definitions

- ☒ Automated Tests
- ☒ Unit Testing
- ☒ **Test-last / test-after**
- ☐ Test-first
- ☐ Code Coverage

Definitions

- ☒ Automated Tests
- ☒ Unit Testing
- ☒ Test-last / test-after
- ☐ **Test-first**
- ☐ Code Coverage

Tests

test_1...

test_2...

test_3...

test_4...

Implementation

```
foo = function() {  
  ...  
}
```

```
bar = function() {  
  ...  
}
```

Definitions

- ☒ Automated Tests
- ☒ Unit Testing
- ☒ Test-last / test-after
- ☒ **Test-first**
- ☐ Code Coverage



I have tests.
They're automatically
repeated.

$$\frac{800 \text{ Lines executed by tests}}{1000 \text{ Lines of code}} = 80\% \text{ Code coverage}$$

Definitions

- ☒ Automated Tests
- ☒ Unit Testing
- ☒ Test-last / test-after
- ☒ Test-first
- ☒ **Code Coverage**