



# Test Driven Development

Lewis England & Erlend Aakre  
[armakuni.com](https://armakuni.com)



# Learning Objectives

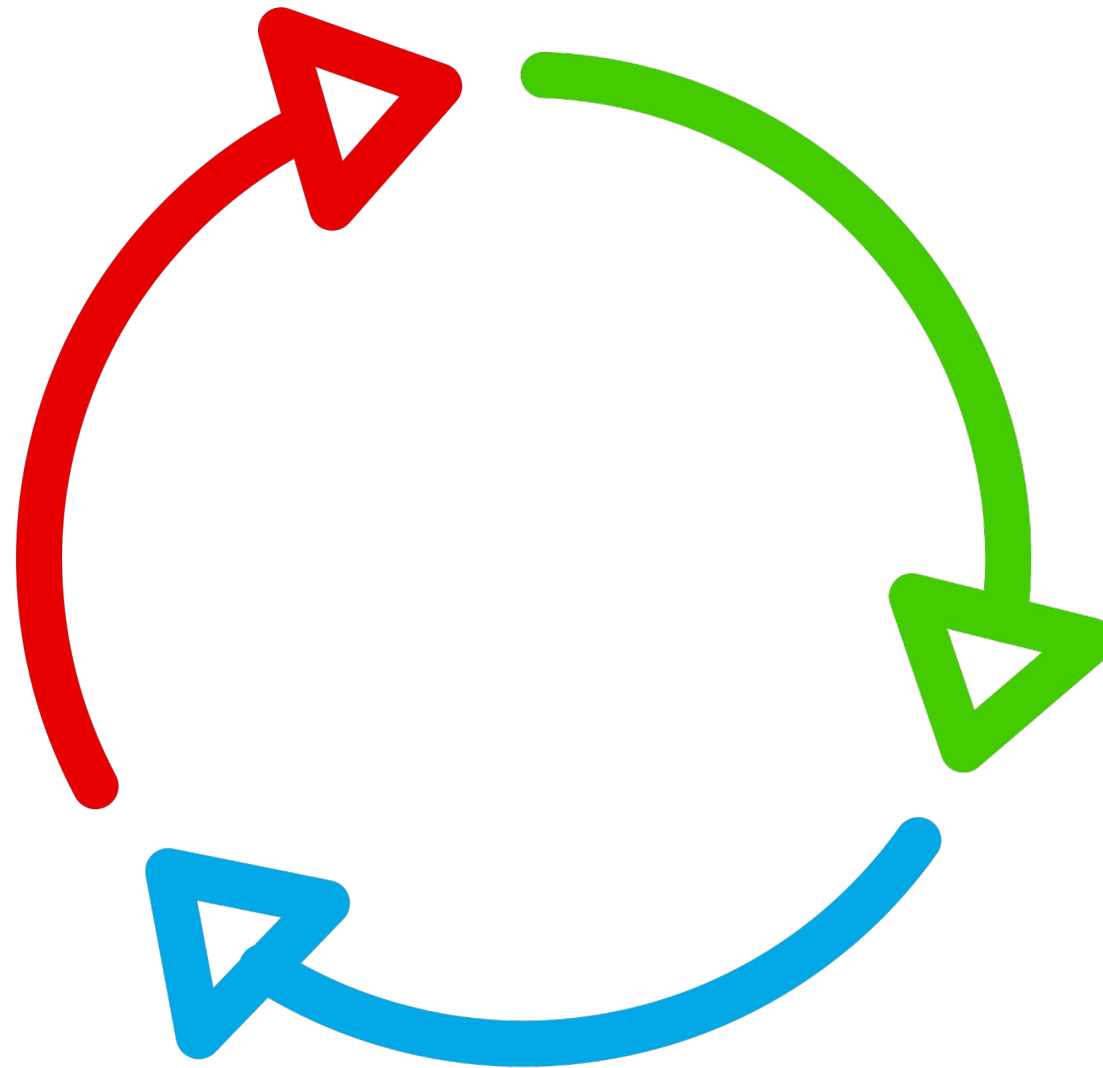


- Describe concept & benefits of Test Driven Development (TDD)
- Describe **red**, **green**, **refactor** cycle
- Use pytest to unit test a Python app



# What is TDD?

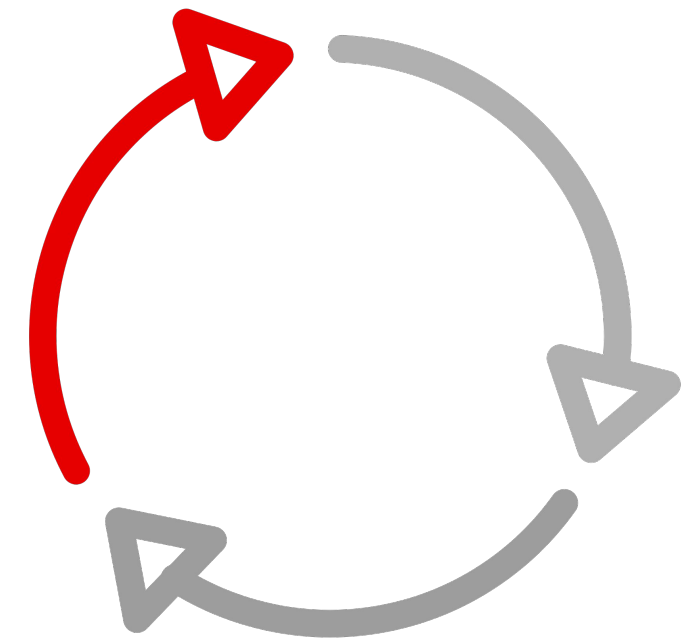
# Red, Green, Refactor Cycle



# The TDD Cycle



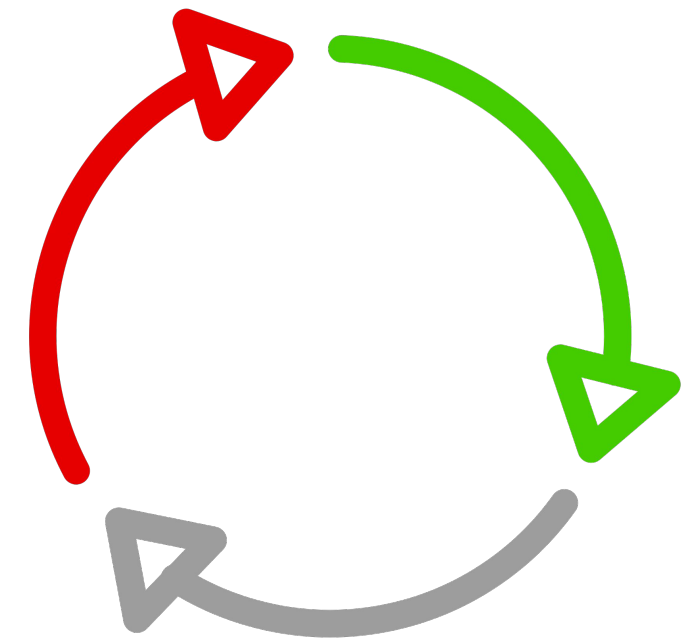
Write a test and check it fails



# The TDD Cycle



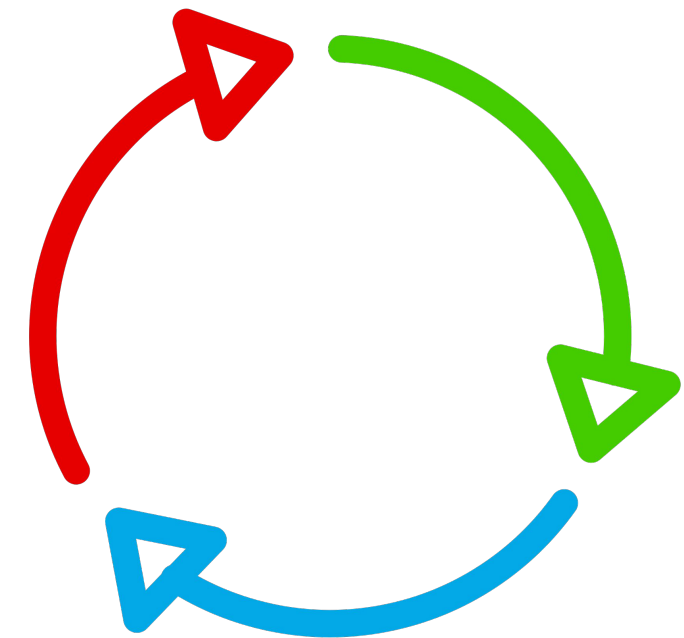
Write a test and check it fails  
Write the code to make it pass



# The TDD Cycle



Write a test and check it fails  
Write the code to make it pass  
Refactor as needed





# Intro: Pytest with Hello World



# Pytest



```
1  # src/hello.py
2  def hello():
3      return "Hello, World!"
4
5  # test/test_hello.py
6  def test_hello():
7      expected = "Hello, World!"
8      actual = hello()
9
10     assert expected == actual
11
```



# Exercise: The OmniThingyParser™

# OmniThingyParser™



- I want it to read a JSON file
- I now want it to read YAML
- Read from `CompanyDbConnector`\*
- Read from `unstableDataProcessor`

\* companyDbConnector is under development. Mock the responses as it's not finished

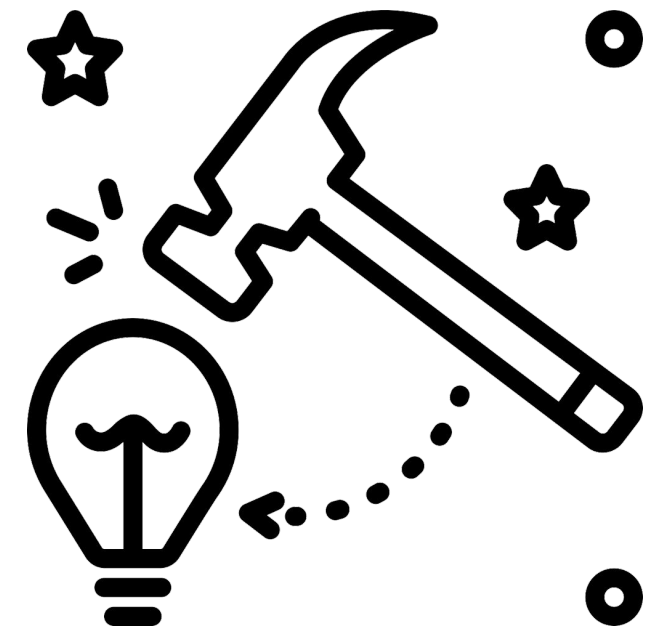


# Benefits of TDD

# Benefits



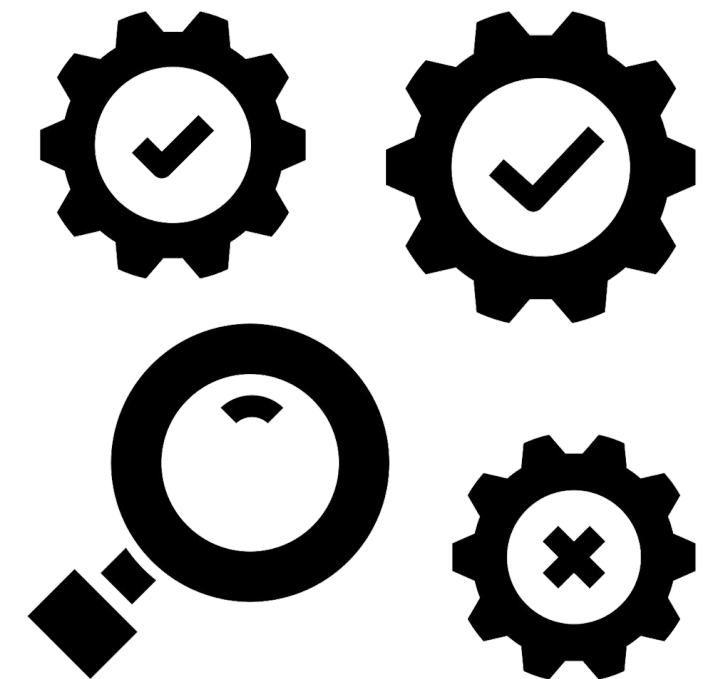
Remove fear of breaking something and not knowing



# Benefits



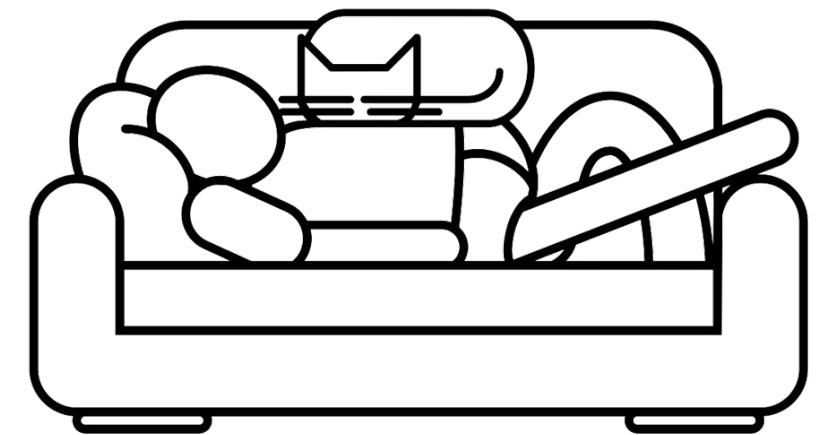
Remove fear of breaking something and not knowing  
Creates a simpler, iterative, more resilient design



# Benefits



Remove fear of breaking something and not knowing  
Creates a simpler, iterative, more resilient design  
Self-documenting code



# Cheat Sheet: Testing



Why Test	Tricks and Tips	The Words You Use Matter	Key Terms
<ul style="list-style-type: none"><li>● Know when you've broken something</li><li>● Improve design</li><li>● Document the code and behaviour</li></ul>	<ul style="list-style-type: none"><li>● Hard to write unit tests are telling you your code is bad</li><li>● Don't add special "test only" functions</li><li>● Test inputs and outputs</li><li>● Make sure you have seen your tests fail at least once</li></ul>	<p>Your tests are part of the documentation for the next developer. Make sure it explains the functionality of the code with clear examples of behaviour and good test naming</p>	<ul style="list-style-type: none"><li>● Unit Test</li><li>● Assertion</li><li>● Expectation</li></ul>



