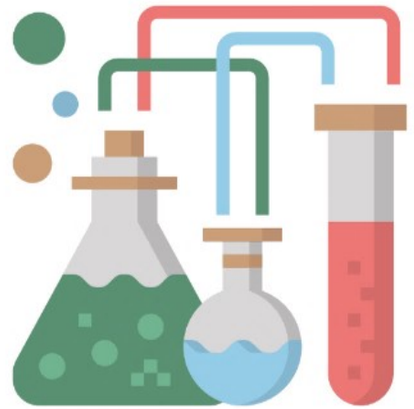
alterra academy

Let's get started!

WHAT IS SOFTWARE TESTING ?

A process of **analyzing a software item** to **detect the differences** between existing and required conditions (i.e., defects) and to **evaluate the features** of the software item.



(Menurut standar ANSI / IEEE 1059)

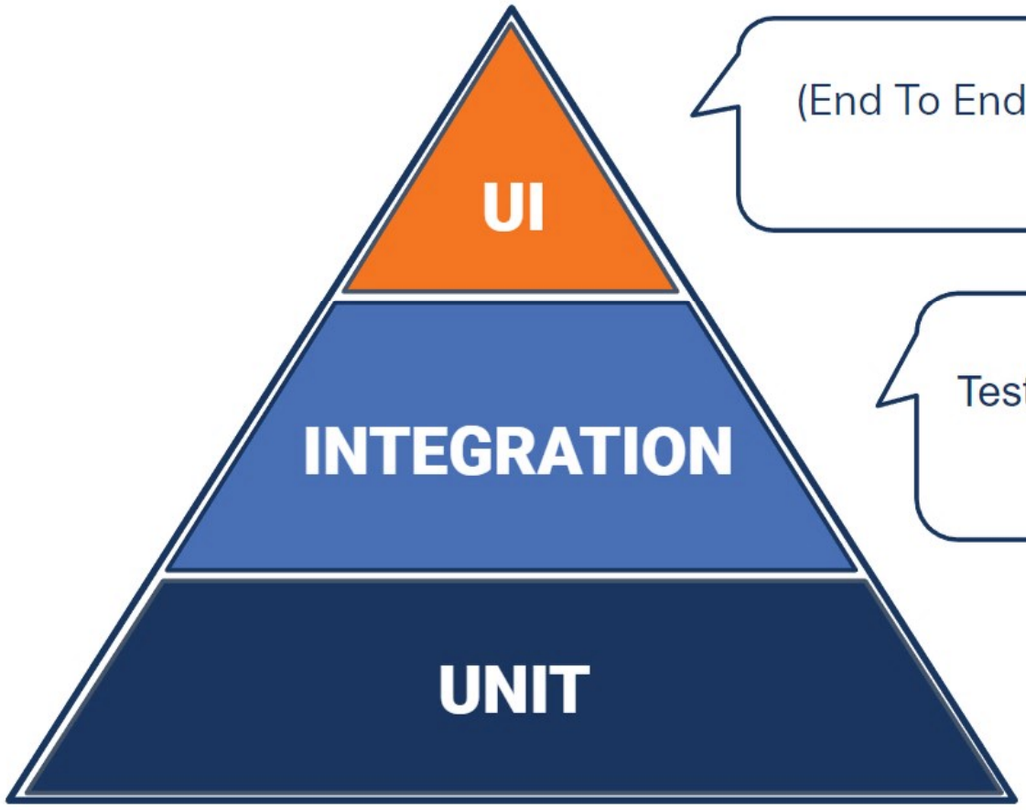
alterra academy

PURPOSE OF TESTING

- Preventing **Regression**
- Confidence Level in **Refactoring**
- Improve **Code Design**
- Code **Documentation**
- Scaling The **Team**

alterra academy

LEVEL OF TESTING



(End To End) Test the interaction between The overall through the user Interface

Tests a specific modules or sub-system through the api

Tests the smallest testable part (single logical operation) of an application through methods

FRAMEWORK

alterra
academy

alterra
academy

Based on your programming language

YOU MUST CHOOSE

unit testing **framework**.

alterra
academy

Framework provides tool,
and structure necessary
to run **Testing**
efficiently.

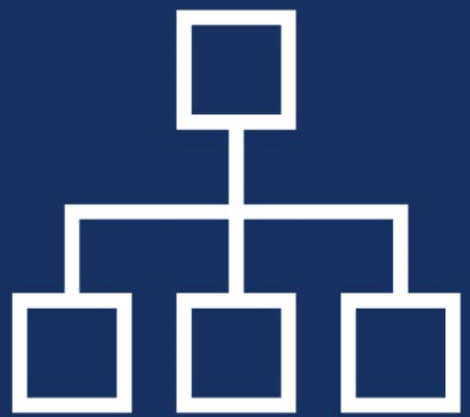
alterra academy

List of

(https://en.wikipedia.org/wiki/List_of_unit_testing_frameworks)

unit testing **framework**.

alterra academy



STRUCTURE

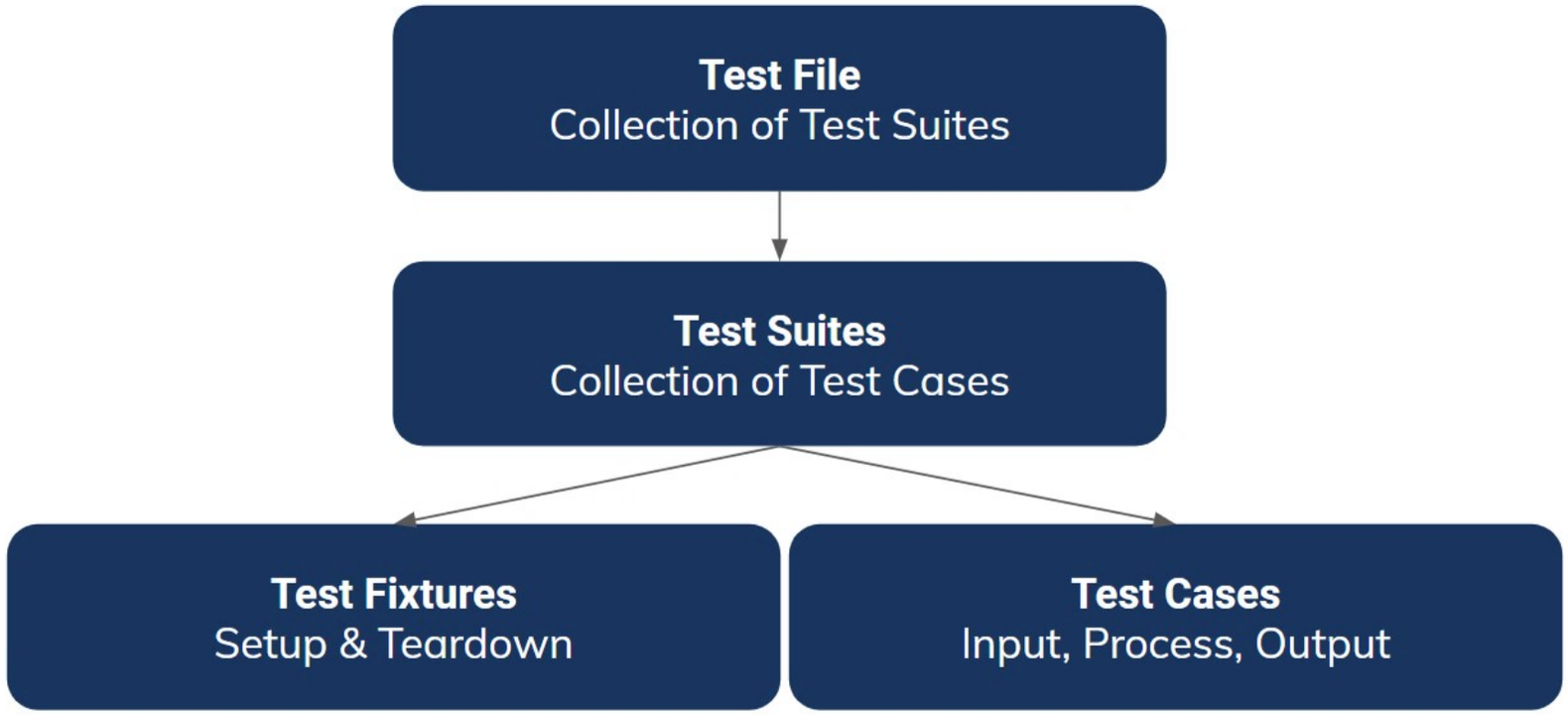
alterra
academy

alterra academy

2 usual pattern :

1. **Centralize** your test file inside tests folder.
2. Save test file **together** with production file.

alterra academy



RUNNER

alterra
academy

alterra academy

Tool that run the test file

Use watch mode

(if there is change in codebase, automatically run test again, make TDD more efficient)

Choose a runner that can run fastest

MOCKING

alterra
academy

alterra
academy

Your test case
need to be
INDEPENDENT

alterra academy

YOU DO NOT TO TEST

- 3rd party modules
- Database
- 3rd party api or other external system
- Object class that you have test in other place

alterra academy

You **need** to **create**

MOCK OBJECT

(n) fake object that simulate behaviour of real object

*) terms : fakes, spies, stubs, mocks

COVERAGE

alterra
academy

Coders need to

make sure

whether they have created enough tests.

alterra
academy

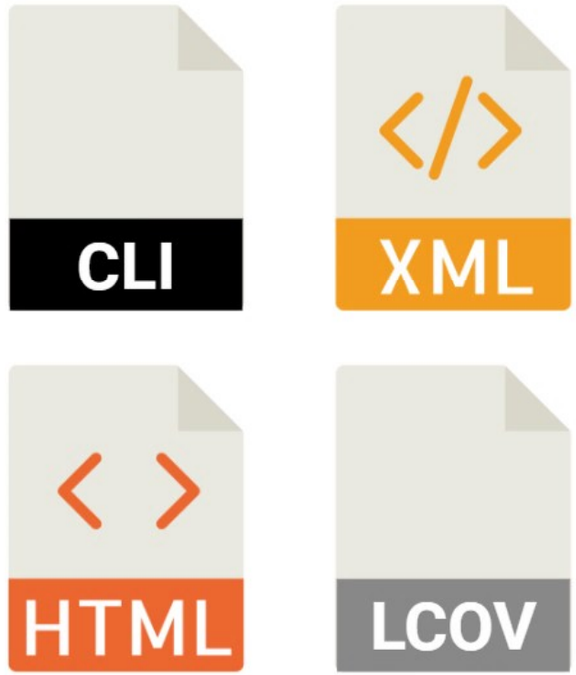
Coverage tool analyzes application code while tests are *running*.

alterra academy

COVERAGE REPORT

- Function
- Statement
- Branch
- Lines

REPORT FORMAT

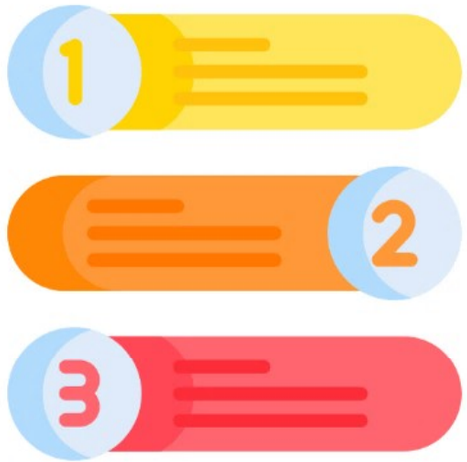


alterra academy

Use format that tool like

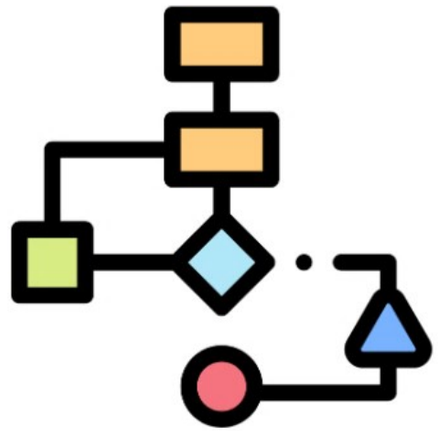
(static code analyzer)

able to **read it**.



Simple steps to test

Create a new test file in Go!



alterra academy

1. name ending library_test.go (e.g. user_test.go)
2. Location File:
   a. same folder, same package
   b. same folder, different package

Create package file

alterra academy

-code editor

```
# lib/calculate/calculate.go
package calculate

func Additon(a, b int) int {
  result := a + b
  return result
}
```

Write a test function

alterra academy

1. Name : **Test & Capitalised Word**
2. Should have the signature func TestXxx(t *testing.T)

alterra academy

-code editor

```
# lib/calculate/calculate_test.go
package calculate

import (
  "testing"
)

func TestAdditon(t *testing.T) {
  if Additon(1, 2) != 3 {
    t.Error("Expected 1 (+) 2 to equal 3")
  }
  if Additon(-1, -2) != -3 {
    t.Error("Expected -1 (+) -2 to equal -3")
  }
}
```

alterra academy

Understand what you just made!

- More than one test function!
- Follow the Rule Testing in Go

alterra

-terminal

```
$ go test ./lib/calculate -cover
```

-terminal

```
$ go test ./lib/calculate
-coverprofile=cover.out && go tool cover
-html=cover.out
```

alterra
academy

Unit Test
REST API
Function

Config

alterra academy

-code editor

```go
var DB *gorm.DB

func InitDB() {

    config := map[string]string{
        "DB_Username": "root",
        "DB_Password": "123ABC4d.",
        "DB_Port":     "3306",
        "DB_Host":     "127.0.0.1",
        "DB_Name":     "training",
    }

    connectionString := fmt.Sprintf("%s:%s@tcp(%s:%s)/%s?charset=utf8&parseTime=True&loc=Local",
        config["DB_Username"],
        config["DB_Password"],
        config["DB_Host"],
        config["DB_Port"],
        config["DB_Name"])

    var e error
    DB, e = gorm.Open(mysql.Open(connectionString), &gorm.Config{})
    if e != nil {
        panic(e)
    }
    InitMigrate()
}

func InitMigrate() {
    DB.AutoMigrate(&models.Users{})
}
```

Controllers DB

alterra academy

-code editor

```go
func GetUsers() (interface{}, error) {
   var users []models.Users

   if e := config.DB.Find(&users).Error; e != nil {
       return nil, e
   }
   return users, nil
}
```

userControllers

alterra academy

-code editor

```go
package controllers

func GetUserControllers(c echo.Context) error {
    users, e := database.GetUsers()
    if e != nil {
        return echo.NewHTTPError(http.StatusBadRequest, e.Error())
    }
    return c.JSON(http.StatusOK, map[string]interface{}{
        "status": "success",
        "users":  users,
    })
}
```

Write a test function for echo

alterra academy

1. Setup Echo (Method, Params, etc)
2. Call function rest api in package controller
3. Create test function

TestUserControllers

alterra academy

-code editor

```
package controllers

func InitEcho() *echo.Echo {
   // Setup
   config.InitDB()
   e := echo.New()

   return e
}

func TestGetUserControllers(t *testing.T) {

   var testCases = []struct {
       name                 string
       path                 string
       expectStatus         int
       expectBodyStartsWith string
   }{
       {
           name:                 "berhasil",
           path:                 "/users",
           expectBodyStartsWith:
"{\"status\":\"success\",\"users\":[",
           expectStatus:         http.StatusOK,
       },
     }
```

-code editor

```
e := InitEcho()
   req := httptest.NewRequest(http.MethodGet, "/", nil)
   rec := httptest.NewRecorder()
   c := e.NewContext(req, rec)

   for _, testCase := range testCases {
       c.SetPath(testCase.path)

       // Assertions
       if assert.NoError(t, GetUserControllers(c)) {
           assert.Equal(t, http.StatusOK, rec.Code)
           body := rec.Body.String()
           // assert.Equal(t, userJSON, rec.Body.String())
           assert.True(t, strings.HasPrefix(body,
testCase.expectBodyStartsWith))

       }
   }
```

alterra
academy

RUN
WITH REPORT
COVERAGE
ALL PACKAGE

-terminal

```
$ go test -v -coverpkg=./...
-coverprofile=profile.cov ./…

$ go tool cover -func profile.cov
```

alterra
academy

“Quality is never an accident; it is always the result of intelligent effort.”

~ John Ruskin

Thankyou !