Automation Testing

©Simplilearn. All rights reserved.

Frameworks of TDD

©Simplilearn. All rights reserved.

A Day in the Life of an Automation Test Engineer

Anna now understands the red-green-refactor and how it works.

Now she needs to be aware of the TDD tools, framework, and environment and how to apply Hamcrest, AssertJ, Kata, and Fizz Buzz to TDD.

To achieve the above, she will learn a few concepts in this lesson that can help her to find a solution to the scenario.

©Simplilearn. All rights reserved.

Learning Objectives

By the end of this lesson, you will be able to:

- Understand the tools, frameworks, and environments used for TDD
- Analyze the integrated development environment
- Describe Hamcrest and AssertJ in details
- Understand TDD Kata and FizzBuzz in depth

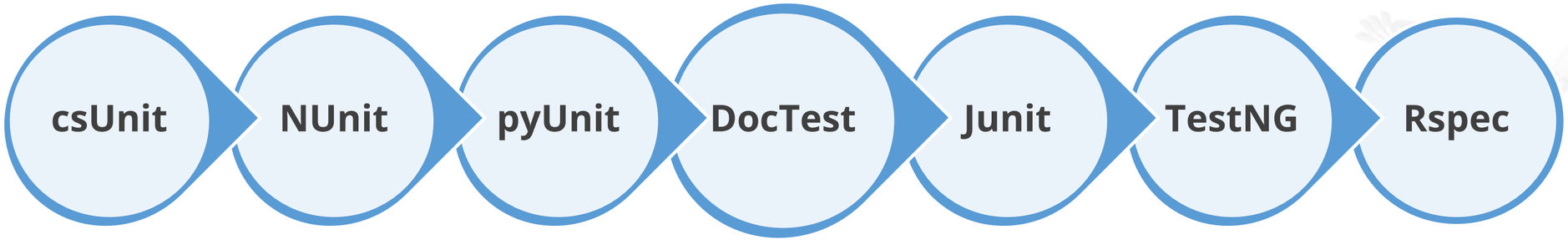
©Simplilearn. All rights reserved.

Tools, Frameworks, and Environment

©Simplilearn. All rights reserved.

TDD Tools

Multiple frameworks that support test driven development are based on different programming languages. A few popular ones are listed below.



©Simplilearn. All rights reserved.

simplilearn

Environment

A test double is a test-specific capability that stands in for a system capability that the UUT relies on, usually a class or function. Test doubles can be introduced into a system in two different ways: link and execution.

When the test double is compiled into the load module and executed to validate testing, it is referred to as link-time substitution. This method is used when running in an environment.

©Simplilearn. All rights reserved.

Test Doubles

There are several varieties of test doubles, each with its level of difficulty:

©Simplilearn. All rights reserved.

simplilearn

FULL STACK

Git

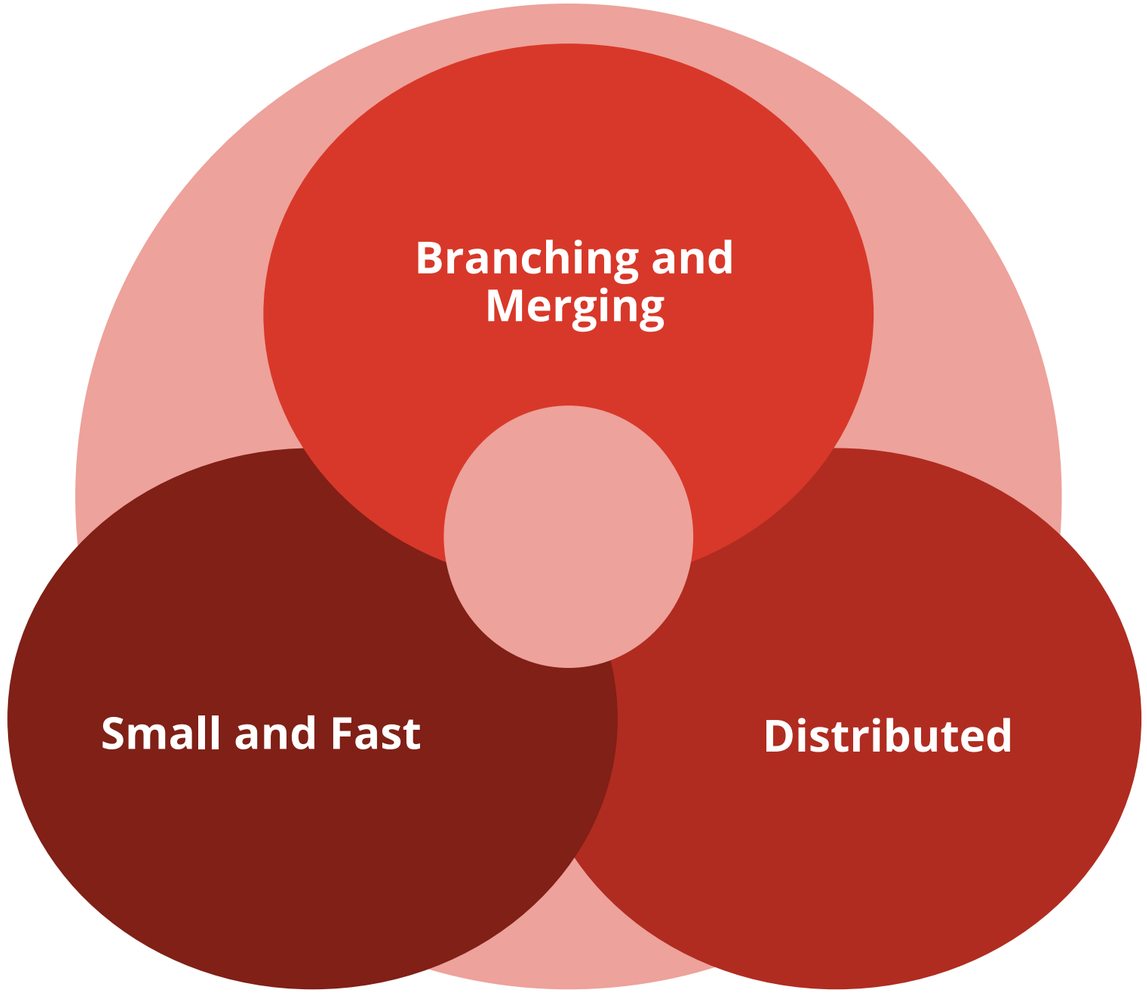
©Simplilearn. All rights reserved.

Introduction to Git

Git is a free and open-source distributed version control system that can handle everything from small to very large projects with ease.

©Simplilearn. All rights reserved.

Features



©Simplilearn. All rights reserved.

simplilearn

Git Commands

The following are some frequently used commands while working with git.

The user name and email address are set using the following command:

```
git config –global user.email “[email address]”
```


©Simplilearn. All rights reserved.

Git Commands

Start a new repository using the following command:

```
git init [repository name]
```


©Simplilearn. All rights reserved.

Git Commands

To get a repository from a URL, use the following command:

```
git clone [url]
```


©Simplilearn. All rights reserved.

simplilearn

Git Commands

To add a file to the staging area, use the following command:

```
git add [file]
```


©Simplilearn. All rights reserved.

Git Commands

Save a copy of the file using the following command:

```
git commit -m "[ Type in the commit message]"
```


©Simplilearn. All rights reserved.

simplilearn

Git Commands

Display the changes in the files using the following command:

```
git diff
```


©Simplilearn. All rights reserved.

Git Commands

Delete a file from the user's current directory using the following command:

```
git rm [file]
```


©Simplilearn. All rights reserved.

Git Commands

To tag a commit, use the following command:

```
git tag [commitID]
```


©Simplilearn. All rights reserved.

Git Commands

Display a list of all the current repositories using the following command:

```
git branch
```

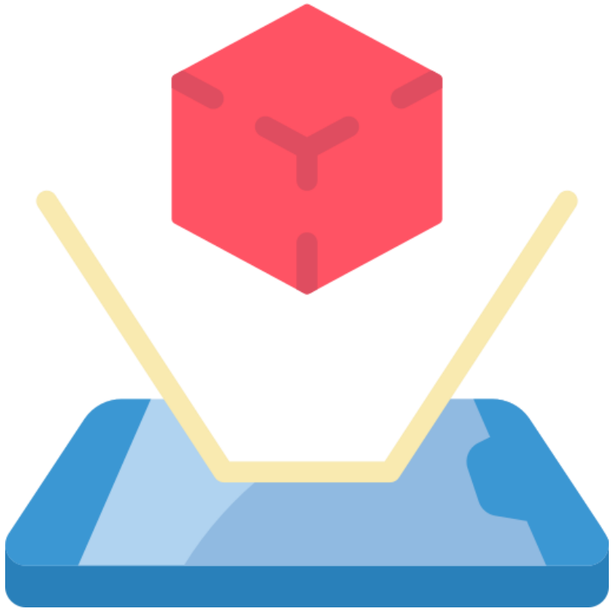

©Simplilearn. All rights reserved.

Virtual Machines

©Simplilearn. All rights reserved.

Introduction to Virtual Machines

A virtual machine is a software application that looks and runs like a different operating system than the one installed on a computer, mobile device, or server.



©Simplilearn. All rights reserved.

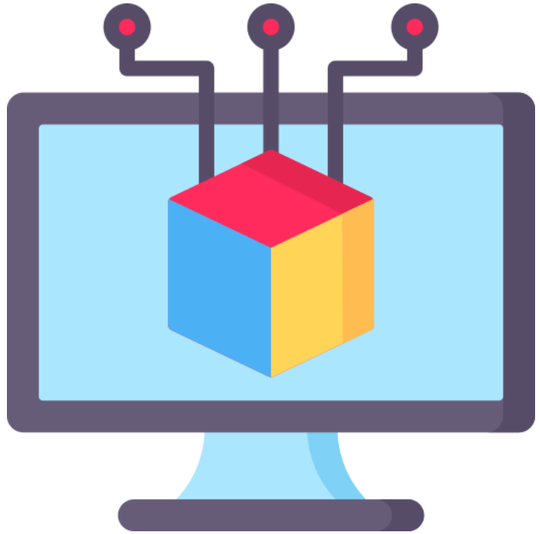
How Does a Virtual Machine Work?

To run a virtual machine (VM), the user must first connect to a host computer or server, which provides the hardware resources. This is known as the host machine.

©Simplilearn. All rights reserved.

Why are Virtual Machines Used?

There are numerous reasons why virtual machines have become a key component of effective IT systems around the world.



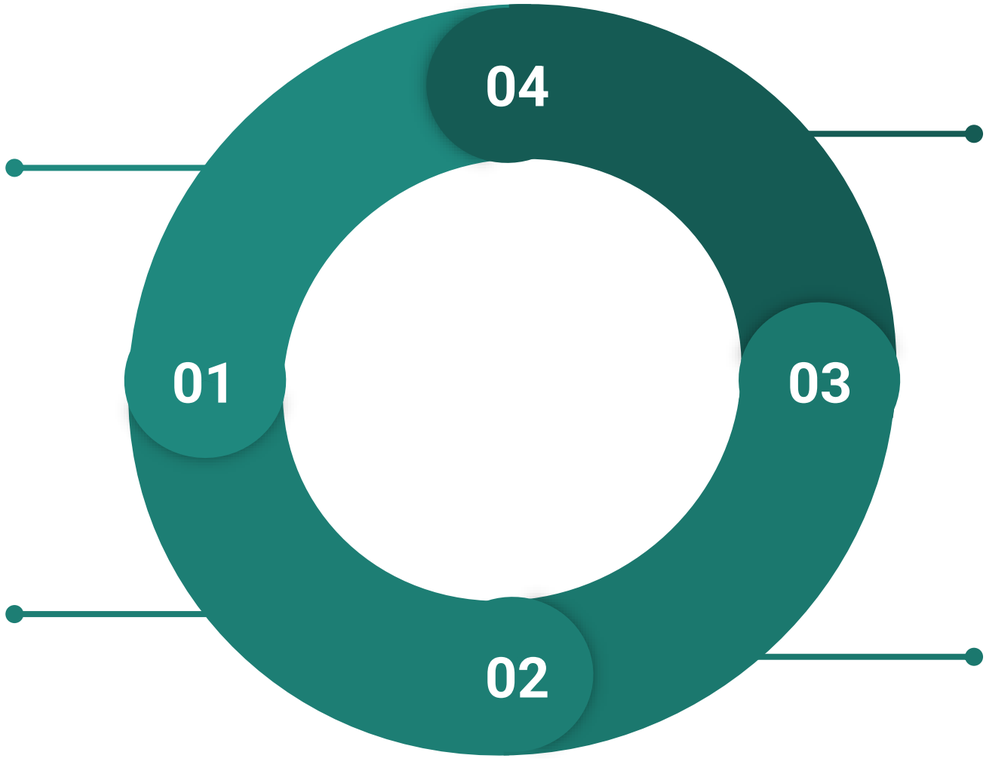
Virtual machines (VMs) enable multisystem applications to run at the same time and in the same location without additional overhead.

©Simplilearn. All rights reserved.

What are Virtual Machines Used For?

Creating and deploying the cloud-based apps

Installing an earlier OS allows users to access virus-infected files or use an old application.



Creating a backup of the user's current operating system

Creating a new environment to make running dev-test scenarios easier and faster for developers

©Simplilearn. All rights reserved.

simplilearn

Introduction to Virtualization

The process of establishing multiple virtual systems on a single server is known as virtualization.

©Simplilearn. All rights reserved.

Benefits of Virtualization

It has more computing power while using fewer resources.

It uses single hardware to run multiple independent systems.

It uses environments that are consistent throughout the continuous integration and delivery (CI/CD) process.

©Simplilearn. All rights reserved.

Different Types of Virtualization

Virtualization is divided into four categories:

- **Server virtualization**
- **Network virtualization**
- **Desktop virtualization**
- **Operating system virtualization**

©Simplilearn. All rights reserved.

Server Virtualization

Because of server virtualization, a single physical server can execute numerous independent functions.

©Simplilearn. All rights reserved.

Network Virtualization

In a virtual environment, network virtualization replicates a network.

©Simplilearn. All rights reserved.

Desktop Virtualization

Desktop virtualization simulates the settings and apps of a desktop device in a virtual environment.

©Simplilearn. All rights reserved.

Operating System Virtualization

A developer can use operating system virtualization to run numerous operating systems on a single machine.

©Simplilearn. All rights reserved.

simplilearn

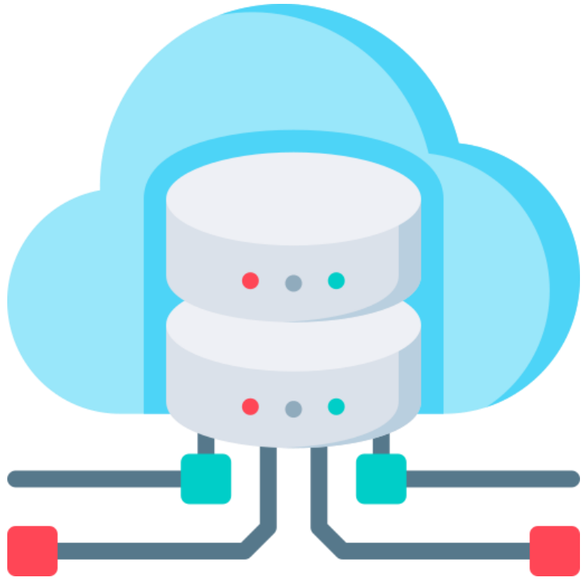
What is DevOps?

DevOps is a collection of processes, beliefs, and tools that allows a team to release software at a rapid pace.

©Simplilearn. All rights reserved.

What Role Does Virtualization Play in DevOps?

In the creation of complex cloud, API, and SOA systems, virtualization in DevOps is important.



Test-driven development (TDD) teams that prefer to start their bug hunting at the API layer will benefit from virtual machines.

©Simplilearn. All rights reserved.

Benefits of DevOps and Virtualization

Cost reduction

Reduce failure rates

Easy backup system

Improved security

©Simplilearn. All rights reserved.

Cost Reduction

Throughout the DevOps pipeline, virtualization saves money.

©Simplilearn. All rights reserved.

Easy Backup System

A group can schedule automated data backups every minute.

©Simplilearn. All rights reserved.

Reduce Failure Rates

Check-in and release failure rates are lower with virtualization. DevOps teams often build automated tests to simulate real-world software usage.

©Simplilearn. All rights reserved.

Improved Security

Virtualization creates environments that are more fault-tolerant, consistent, and predictable, allowing for better configuration control, safety assurance, and cybersecurity.

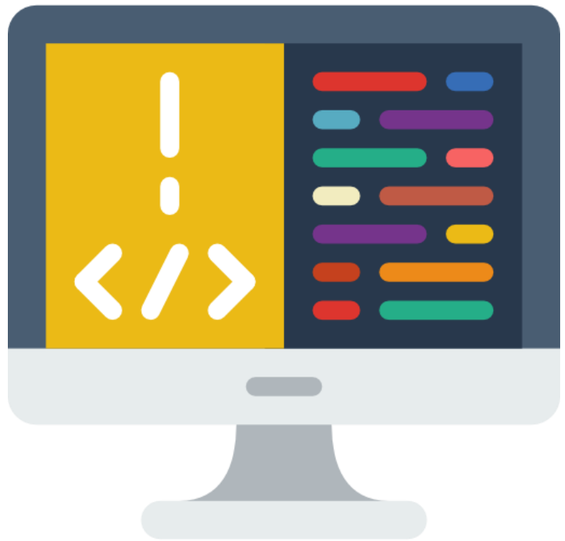
©Simplilearn. All rights reserved.

Build Tools

©Simplilearn. All rights reserved.

Introduction to Build Tools

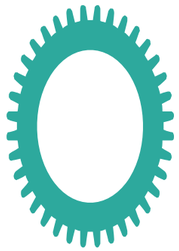
Build tools are programs that automate the process of turning source code into executable applications.



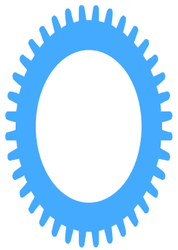
©Simplilearn. All rights reserved.

TDD Tools

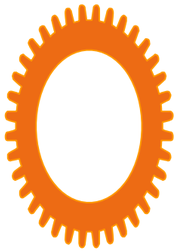
There are various tools available for testing and enhancing the software system's overall design and implementation. The following are some of the most commonly used testing tools:



JUnit



JMeter



Mockito

©Simplilearn. All rights reserved.

JUnit

JUnit is a Java-based unit-testing framework.

©Simplilearn. All rights reserved.

JMeter

The Apache JMeter program is a free and open-source software that is made entirely of Java and is used to load tests and measure performance.

It was originally created to test web applications, but it, has now been expanded to include other test functions.

©Simplilearn. All rights reserved.

Mockito

Mockito enables Test-Driven Development (TDD) programmers to build and test double objects (mock objects) in automated unit tests (TDD).

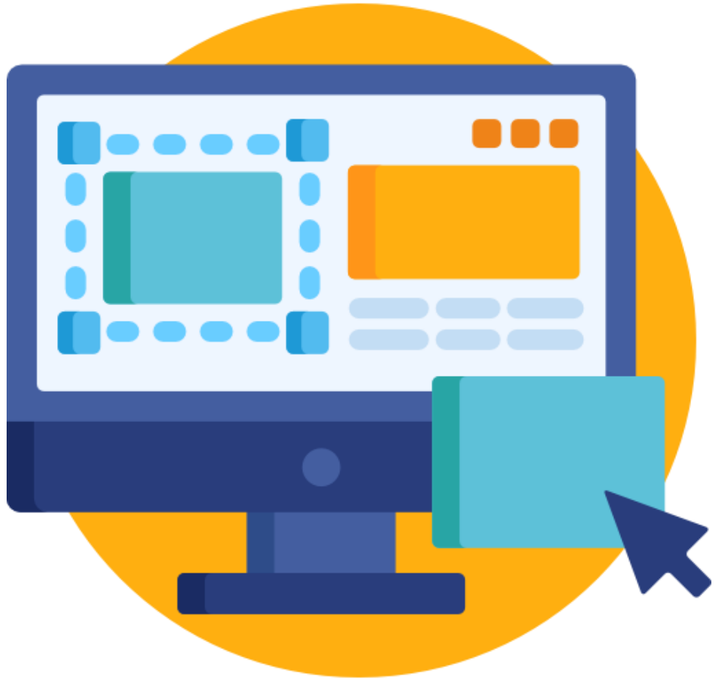
©Simplilearn. All rights reserved.

The Integrated Development Environment

©Simplilearn. All rights reserved.

Introduction to IDE

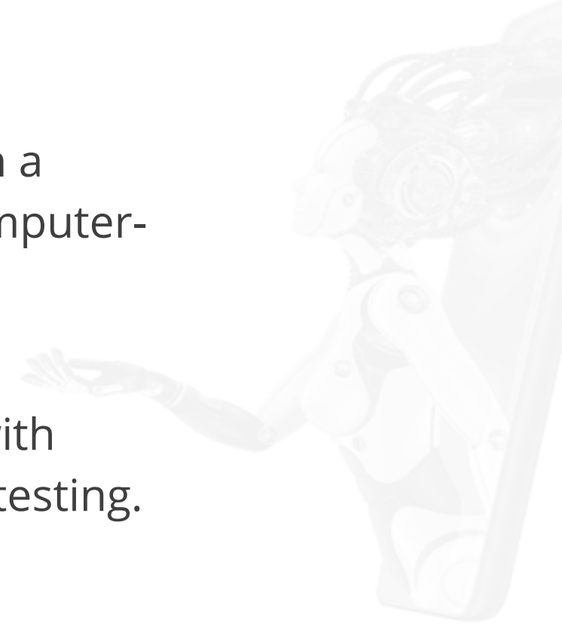
Integrated development environments (IDE) are programs that make it easier to create other programs.



©Simplilearn. All rights reserved.

Benefits

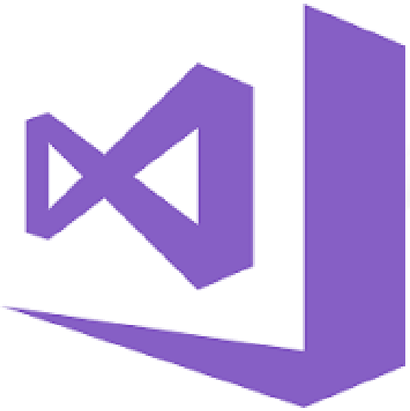
Code editor	These editors, which are designed for writing and editing source code, differ from text editors.
Compiler	These convert source code written in a human-readable language into a computer-executable format.
Debugger	These are used to help developers with debugging their applications during testing.
Build automation tools	These can be used to automate more typical development processes to save time.



©Simplilearn. All rights reserved.

IDE Examples

A text editor, a project editor, a toolbar, and an output viewer are all included in an IDE.



©Simplilearn. All rights reserved.

Unit Testing Framework

©Simplilearn. All rights reserved.

Introduction to Unit Testing

Unit testing is a method of testing the smallest block of code that can be logically isolated in a system, referred to as a unit.

A unit is the smallest tested component of any software with one or more inputs, and one output.

©Simplilearn. All rights reserved.

Framework for C#

Framework used for C# is:

NUnit

The framework used for C# is a part of the xUnit family.

©Simplilearn. All rights reserved.

Example

An example of NUnit:

```
namespace TestingExample
{
      public class Calc
      {
            public int add(int p, int q)
            {
                  int x = p+q;
                  return x;
            }
      }
}
```


©Simplilearn. All rights reserved.

simplilearn

Example

An example of NUnit:

```
using NUnit.Framework;
using TestingExample;
namespace NUnitProject
{
      public class ClacTest
      {
            public void addMethod()
            {
                  ClacTest add = new
CalcTest();
                  int res = add.ADD(20,60);
```


©Simplilearn. All rights reserved.

simplilearn

Example

An example of NUnit:

```
Assert.That(res, Is.EqualTo(80));
 }
[TestCase(20, 35, 55)]
[TestCase(10, 55, 65)]
[TestCase(10, 60, 70)]
```


©Simplilearn. All rights reserved.

simplilearn

Example

An example of NUnit:

```
public void addMethod(int no1, int no2, int expect)
        {
            CalcTest add = new CalcTest();
            int res = add.Add(no1, no2);
            Assert.AreEqual(expect, res);
        }
    }
}
```


©Simplilearn. All rights reserved.

simplilearn

Example

An example of NUnit:

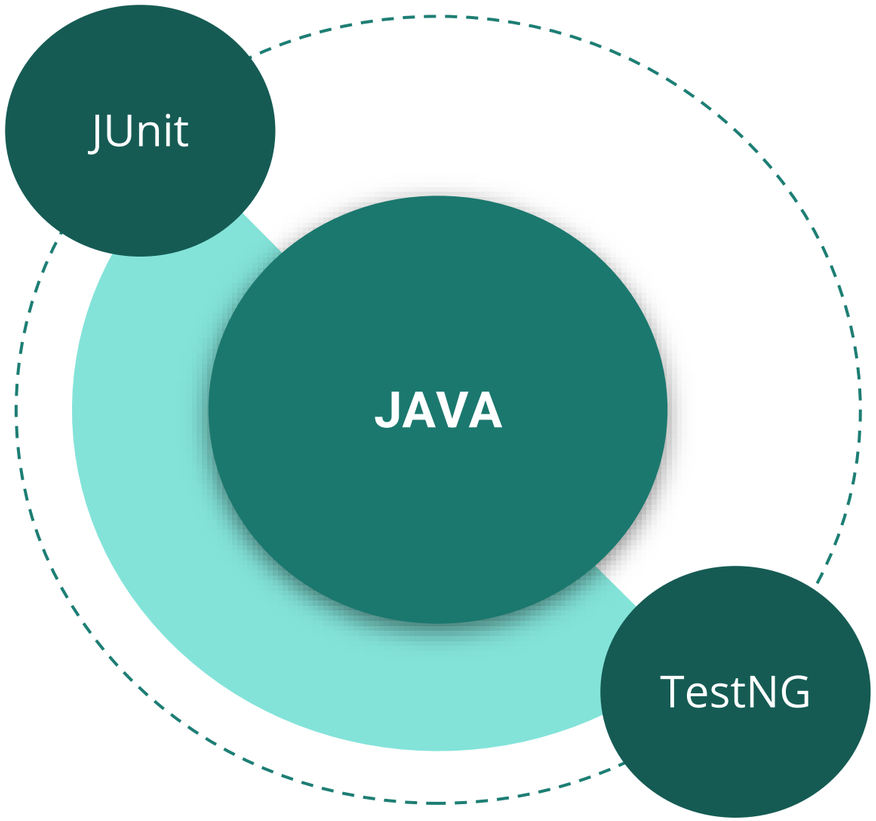
```
[TestCase(20, 35, 55)]
[TestCase(10, 55, 65)]
[TestCase(10, 60, 70)]
```


©Simplilearn. All rights reserved.

simplilearn

Frameworks for Java

The frameworks used for Java are:

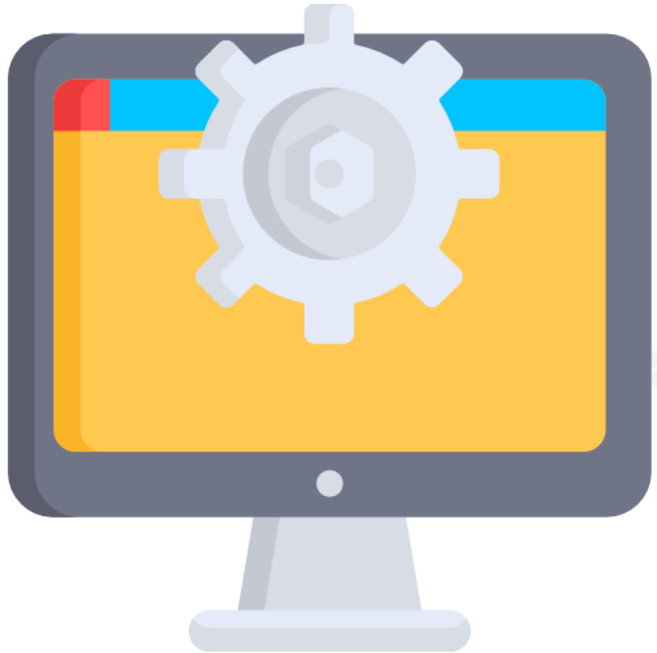


©Simplilearn. All rights reserved.

TestNG

To add a TestNG to Eclipse, follow these steps:

- Install a new software by opening Eclipse software.
- Visit testng.org and select the link for the project repository.
- By choosing the confirm button, verify the installation process.
- Accept the terms and conditions.



©Simplilearn. All rights reserved.

TestNG

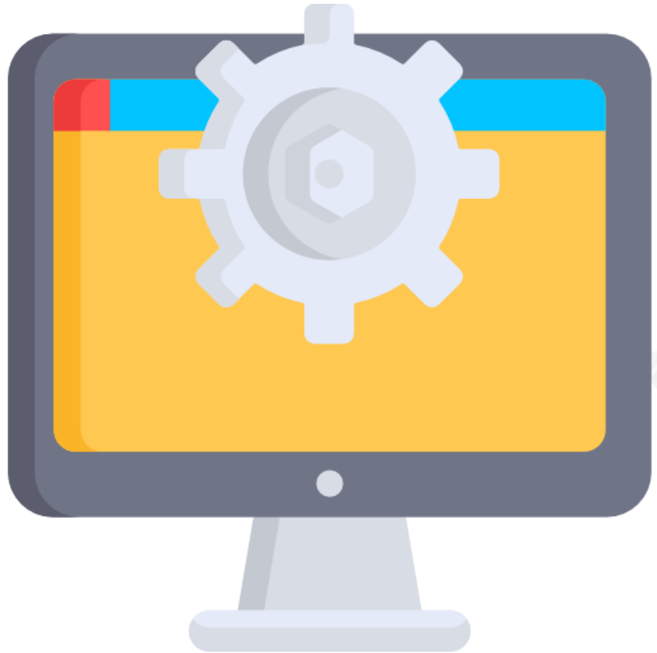
To add a TestNG to Eclipse, follow these steps:

Following the acceptance of the licensing agreement, wait for installation.

Whenever a security alert comes, just select **Install Anyway**.

Select the restart option.

Verify that TestNG for Eclipse has been installed effectively after the restart.



©Simplilearn. All rights reserved.

Framework for C or C++

The framework used for C or C++ is:

Embunit

Embunit is a unit testing tool for software applications written in C or C++. It is intended for both developers and testers.

©Simplilearn. All rights reserved.

Example

An example of Embunit:

```
# include “person.h”

#include “StubEnums.h”

#include “embunit.h”

#include “person_ts.h”


void person_ts::float_exm()

{

      embunit_Case(4);

      embunit_step(1);
```


©Simplilearn. All rights reserved.

simplilearn

Example

An example of Embunit:

```
try
{
      float a(5.1F);
      float b(5.1001F);
      embunit_info("embunit_IsAlmostEqual(a, b, -3)");
      embunit_Test(embunit_IsAlmostEquals(a, b, -3));

   embunit_step(2)
   float c(5.2F);
   embunit_info("!embunit_IsAlmostEqual(a, b, -3)");
```


©Simplilearn. All rights reserved.

Example

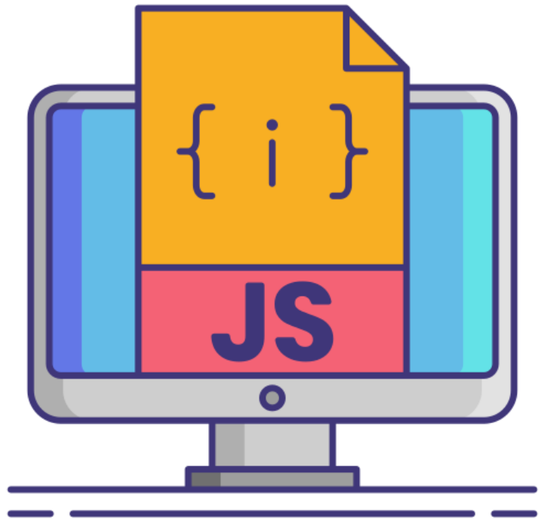
An example of Embunit:

```
embunit_Test(!embunit_IsAlmostEquals(a, b, -3));
}
   catch(...)
   {
      //If the user gets here, something has been thrown
   unexpectedly.
      embunit_AbortTest();
   }
}
```


©Simplilearn. All rights reserved.

simplilearn

Frameworks for JavaScript



HtmlUnit

HtmlUnit is an open-source unit testing framework that supports JavaScript and supports graphical user interface elements such as forms, links, and tables.

©Simplilearn. All rights reserved.

Example

An example of HtmlUnit:

```
final WebClient webClient = new WebClient();

webClient.getOptions.setThrowExceptionOnScriptError(false
);
```


©Simplilearn. All rights reserved.

Example

An example of HtmlUnit:

```
WebClient webClient = new
WebClient(BrowserVersion.FIREFOX, false, null, -1);
```


©Simplilearn. All rights reserved.

Example

An example of HtmlUnit:

```
final WebClient webClient = new
WebClient(BrowserVersion.FIREFOX);
 webClient.getOptions.setJavaScriptEnabled(false);
 ....
 webClient.getOptions.setJavaScriptEnabled(true);
```


©Simplilearn. All rights reserved.

Benefits

Detect flaws in the early stages of development.

Unit tests ensure that refactoring is done safely.

©Simplilearn. All rights reserved.

Benefits

- Provides documentation
- Performs less regression testing

©Simplilearn. All rights reserved.

simplilearn

Hamcrest and AssertJ

©Simplilearn. All rights reserved.

Introduction to Hamcrest

Hamcrest is a Java programming language framework that helps in the creation of software tests.

It allows match rules to be expressed declaratively by generating customized assertion matches (Hamcrest is an anagram of matches).

©Simplilearn. All rights reserved.

simplilearn

Example

Consider the following Hamcrest assertThat method example:

```
import org.junit.jupiter.api.Test;
import static org.hamcrest.MatcherAssert.assertThat;
import static org.hamcrest.Matcher.*;

public class AlphabetTest{
   @Test
   public void testEquals(){
   Alphabet1 theAlphabet = newAlphabet1 (“A”);
   Alphabet1 myAlphabet = newAlphabet1 (“A”);
      assertThat(theAlphabet, equalTo(myAlphabet));
      }
}
```


©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

- **Core**
  - anything
  - describedAs
  - is
- Logical
- Object
- Beans

©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

- Core
- **Logical**
- Object
- Beans

- alloff
- anyoff
- not

©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

- Core
- Logical
- **Object**
- Beans

- equalTo
- hasToString
- instanceOf

©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

- Core
- Logical
- **Object**
- Beans

- notNullValue
- sameInstance

©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

- Core
- Logical
- Object
- **Beans**
  - hasProperty

©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

Collections

Number

Text

array

hasEntry

hasItem

©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

- Collections
- **Number**
  - closeTo
  - greaterThan
- Text

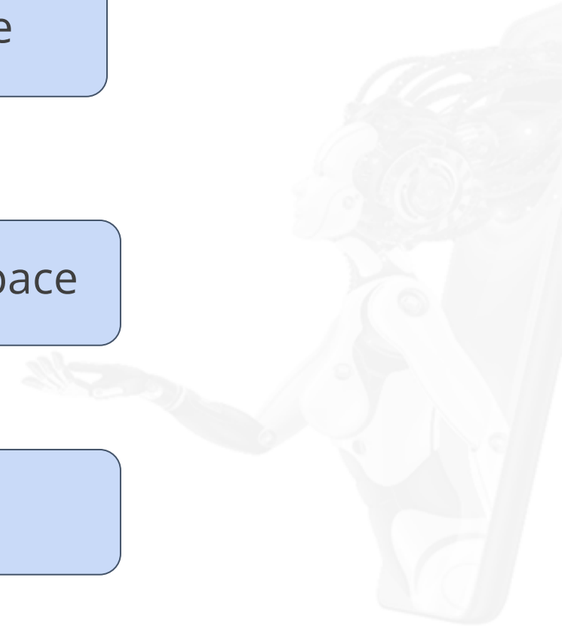
©Simplilearn. All rights reserved.

Common Matchers

The common matchers are as follows:

- Collections
- Number
- **Text**

- equalToIgnoringCase
- equalToIgnoringWhiteSpace
- containsString



©Simplilearn. All rights reserved.

Example

Consider the following Hamcrest example:

```
assertThat(theAlphabet, equalTo(myAlphabet));

assertThat(theAlphabet, is(equalTo(myAlphabet)));

assertThat(theAlphabet, is(myAlphabet));
```


©Simplilearn. All rights reserved.

simplilearn

Introduction to AssertJ

AssertJ is a Java library that provides a comprehensive set of assertions and helpful error messages, increases test code readability, and is designed to be extremely simple to use within your preferred IDE.

©Simplilearn. All rights reserved.

Supported Types Assertions

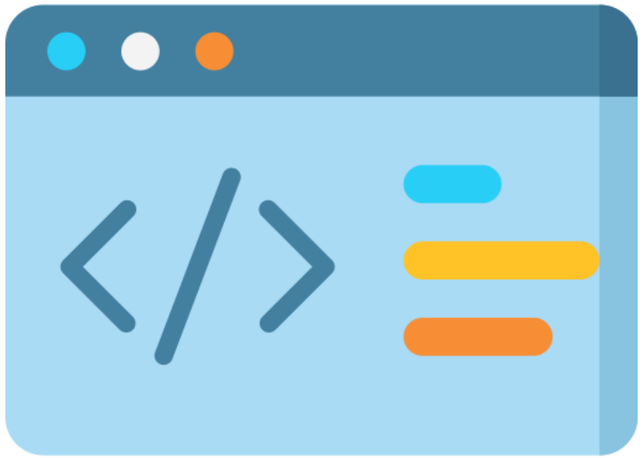
The types of assertions that AssertJ supports are listed below:

BigDecimal	BigInteger
CharSequence	Class
Date	File
Future	InputStream
Iterable	Iterator
List	Map
Object	Object[] and Object[][]
Predicate	Stream

©Simplilearn. All rights reserved.

simplilearn

Primitive Types



Primitive types:

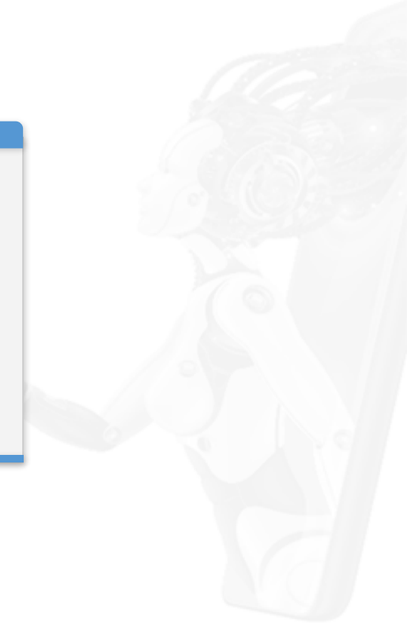
- short or Short
- int or Integer
- long or Long
- byte or Byte
- char or Character
- float or Float
- double or Double

©Simplilearn. All rights reserved.

assertThat Method

The assertThat method is the base method for AssertJ assertions, followed by the assertion:

```
Date today = new Date();
assertThat(birthday).isBefore(today);
```



©Simplilearn. All rights reserved.

assertThat Method

Consider the following example for the asserThat method:

```
List<String> list = new ArrayList<>();
assertTrue(list.contains("pqr"));
->
java.lang.AssertionError at ...

assertThat(list).contains("pqr");
->
java.lang.AssertionError:
    Expecting:
         <[]>
        to contain:
         <["pqr"]>
        but could not find:
         <["pqr"]>
```

©Simplilearn. All rights reserved.

simplilearn

Gradle

Add the following Gradle dependency:

```
testImplementation 'org.assertj:assertj-core:3.21.1'
```


©Simplilearn. All rights reserved.

Maven

Add the following Maven library:

```
<dependency>
    <groupId>org.assertj</groupId>
    <artifactId>assertj-core</artifactId>
    <version>3.21.1</version>
    <scope>test</scope>
</dependency>
```


©Simplilearn. All rights reserved.

simplilearn

Code Coverage Tools

©Simplilearn. All rights reserved.

Introduction to Code Coverage Tools

Following are some popular code coverage tools:

Parasoft JTest

Parasoft Jtest is one of the items in the collection of Parasoft testing tools

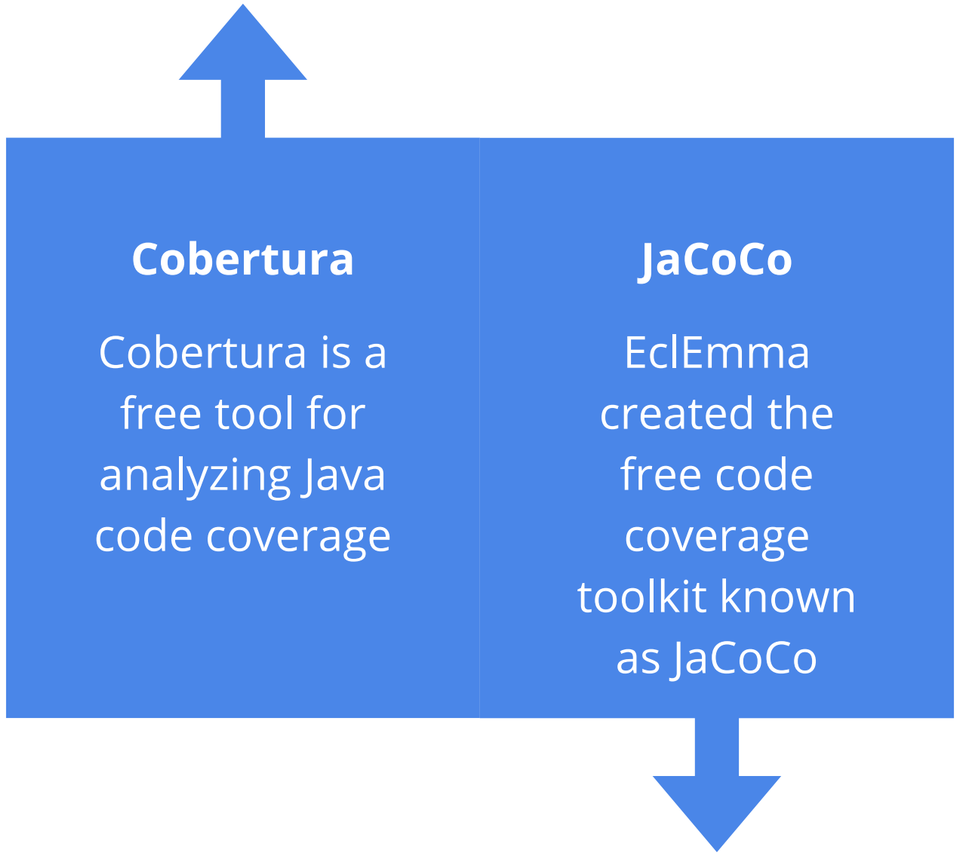
Testwell CTC++

Testwell CTC++ is a dependable Java, C, C++, and C# code coverage and analysis tool

©Simplilearn. All rights reserved.

Introduction to Code Coverage Tools

Following are some popular code coverage tools:



©Simplilearn. All rights reserved.

simplilearn

Introduction to Code Coverage Tools

Following are some popular code coverage tools:

CodeCover

An open-source and extendable CodeCover tool used to test the code coverage of Java programs

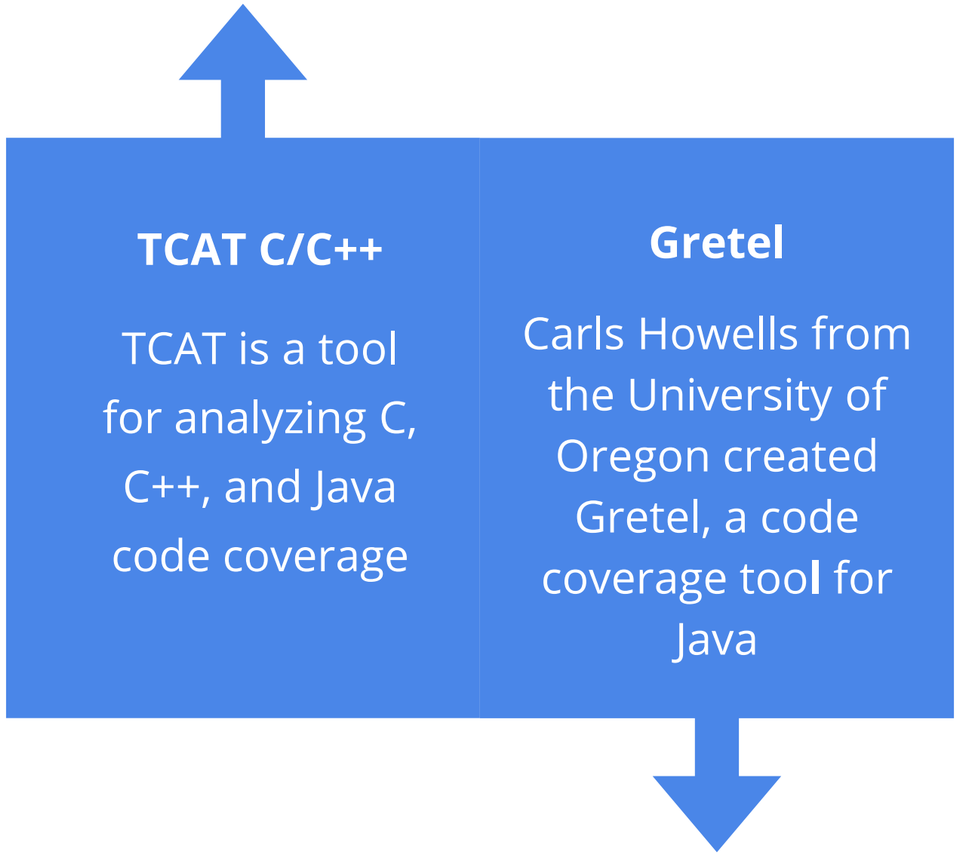
EMMA

A popular open-source method for measuring code coverage in Java projects is called Emma

©Simplilearn. All rights reserved.

Introduction to Code Coverage Tools

Following are some popular code coverage tools:



©Simplilearn. All rights reserved.

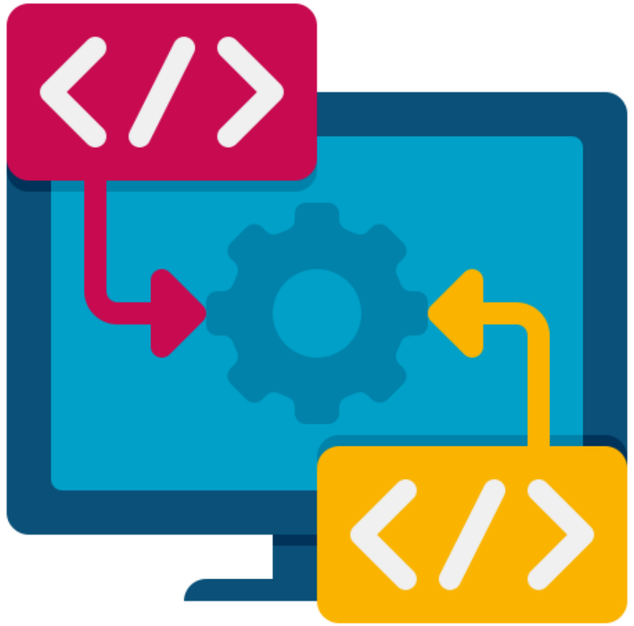
simplilearn

Mocking Frameworks

©Simplilearn. All rights reserved.

Introduction to Mocking Framework

Mocking frameworks generate replacement objects like the stubs and the mocks.



©Simplilearn. All rights reserved.

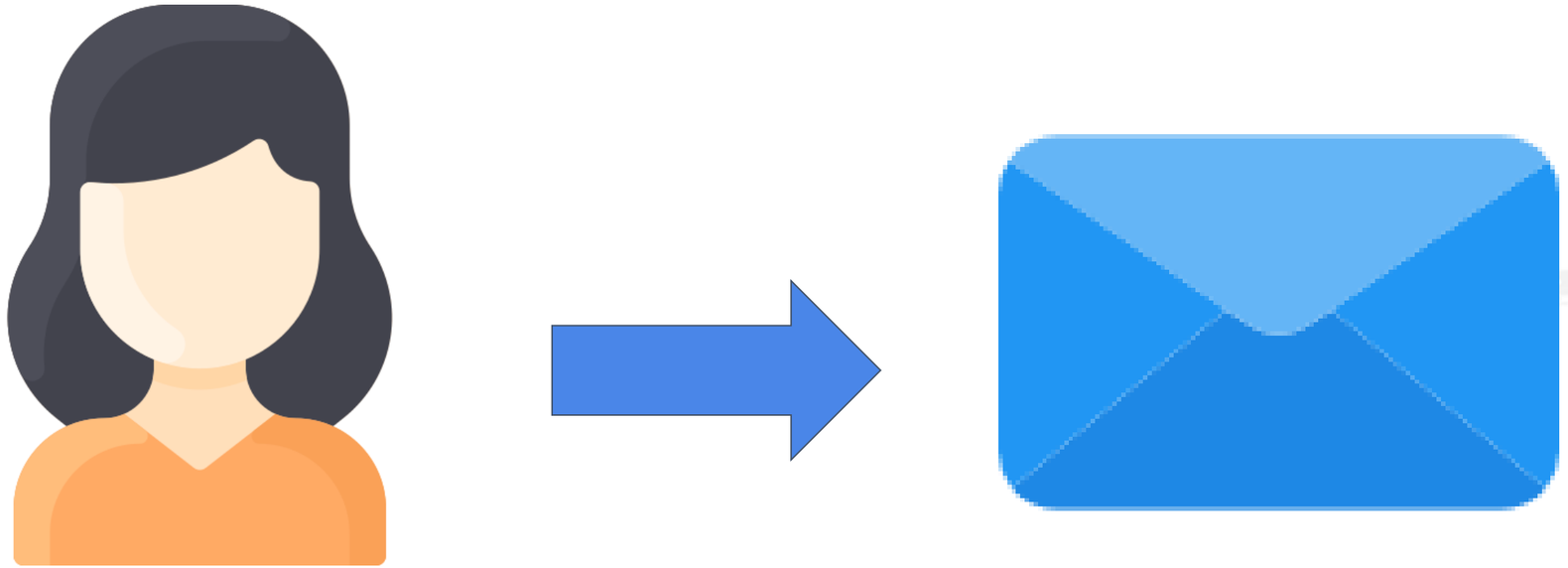
What is a Test Runner?

Test runners implement unit tests. Most unit testing frameworks feature test runners and vary from simple command-line runners to graphical interfaces.

©Simplilearn. All rights reserved.

Example

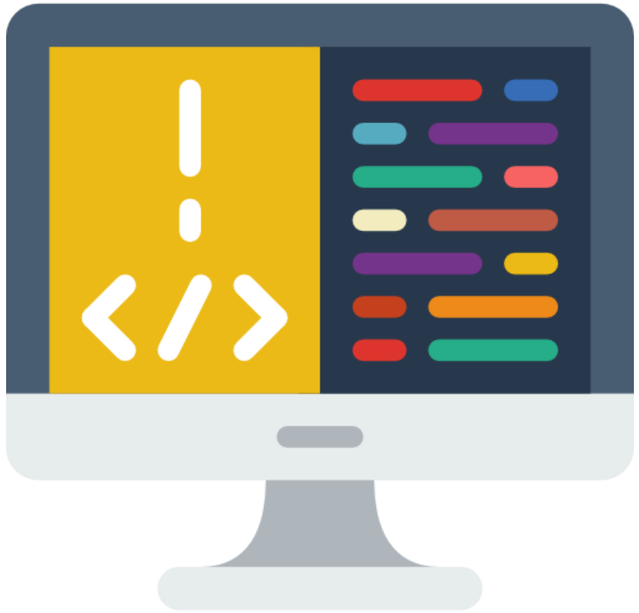
The following example includes sending a notification:



©Simplilearn. All rights reserved.

Mock testing vs Traditional Unit Testing

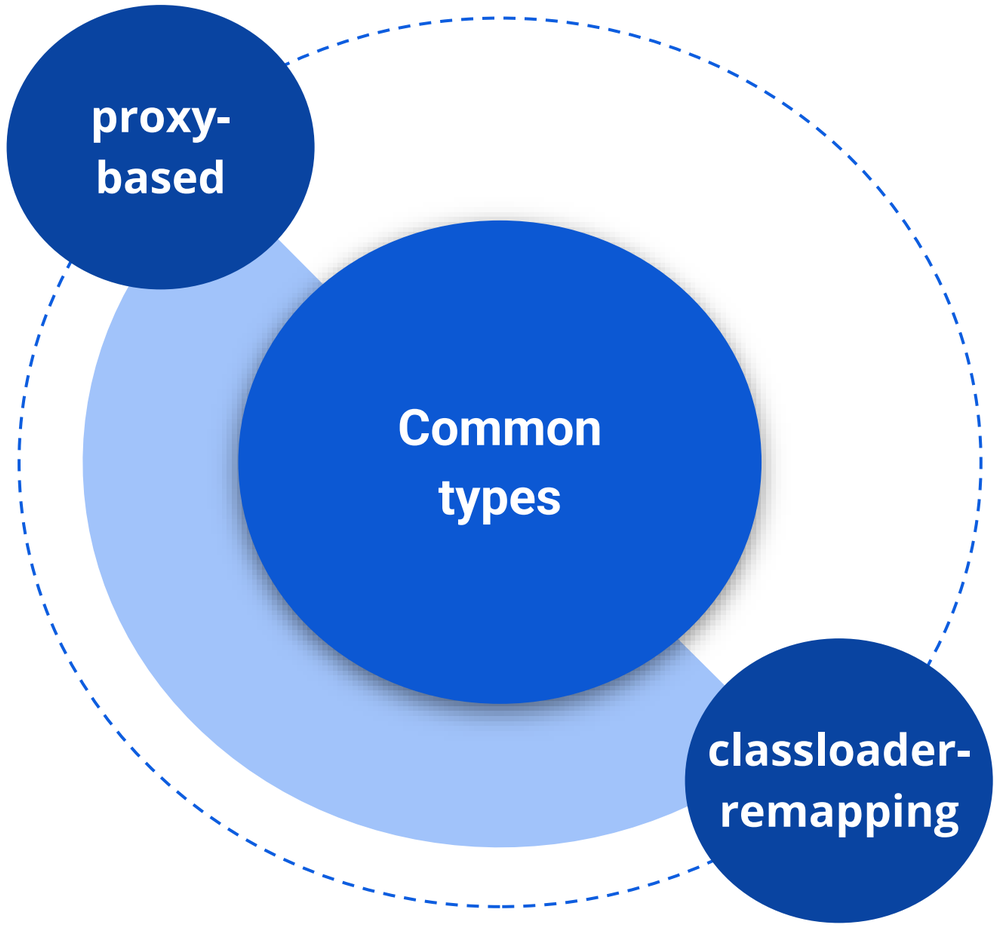
In mock testing, no assertions are needed from the unit tests themselves.



©Simplilearn. All rights reserved.

Types of Mocking

The following are common types of mocking:



©Simplilearn. All rights reserved.

simplilearn

TDD Kata

©Simplilearn. All rights reserved.

Introduction to Kata

Through practice and repetition, a Code Kata is a programming exercise that aids developers in improving their skills.

©Simplilearn. All rights reserved.

Kata and TDD

The best way to practice Test Driven Development is via Katas.

©Simplilearn. All rights reserved.

Example

Example of Kata:

```
public class Kata1
{
    public static string RepeatStr(string p, int
q)
    {
        return "";
    }
}
```


©Simplilearn. All rights reserved.

simplilearn

TDD FizzBuzz

©Simplilearn. All rights reserved.

How to Use TDD Fizz Buzz?

Example of TDD using Fizz Buzz:

```
import java.util.*;
class FizzBuzz
{
    public static void main(String[] args)
    {
        int x = 50;
```


©Simplilearn. All rights reserved.

How to Use TDD Fizz Buzz?

Example of TDD using Fizz Buzz:

```
for (int i=1; i<=x; i++)
    {
            if (i%15==0)
          System.out.println("FizzBuzz1");
```


©Simplilearn. All rights reserved.

simplilearn

How to Use TDD Fizz Buzz?

Example of TDD using Fizz Buzz:

```
else if (i%5==0)
    System.out.println("Buzz");
'Fizz'
```


©Simplilearn. All rights reserved.

How to Use TDD Fizz Buzz?

Example of TDD using Fizz Buzz:

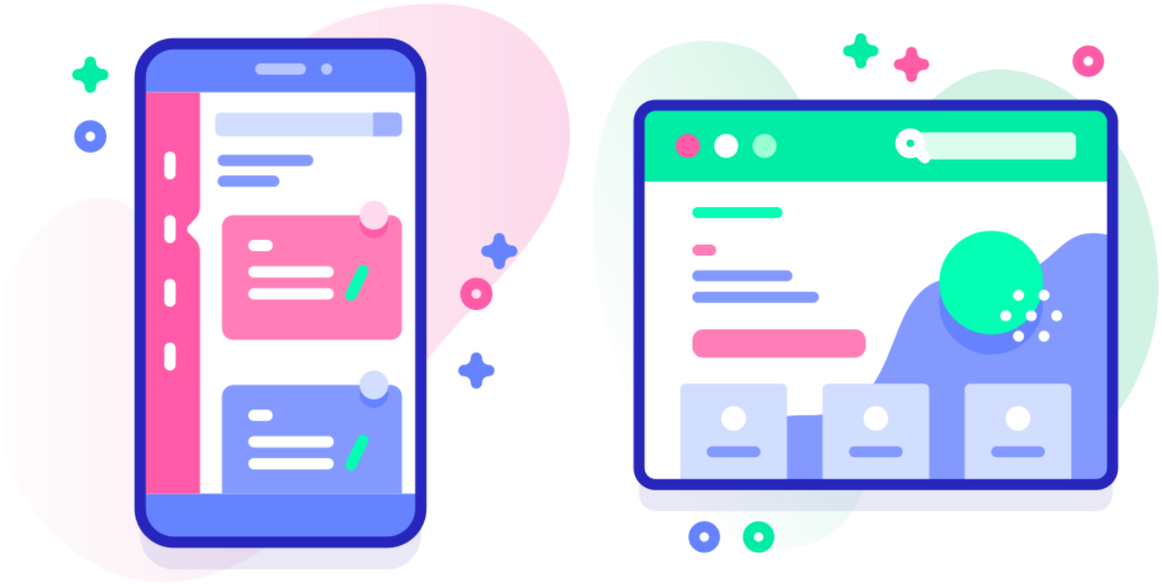
```
            else if (i%3==0)
                  System.out.println("Fizz");


              else // print the numbers
                  System.out.println(i);
          }
      }
}
```


©Simplilearn. All rights reserved.

User Interface Testing

©Simplilearn. All rights reserved.

Introduction to User Interface



Developers and testers are progressively prioritizing UI testing (User Interface testing) as a crucial component of the development plan since UI design and functionality may make or break a piece of software.

©Simplilearn. All rights reserved.

Scope of UI Testing

- Data type errors
- Field widths
- Navigational elements
- Progress bars
- Type-ahead
- Table scrolling
- Error logging
- Menu items
- Working shortcuts

©Simplilearn. All rights reserved.

Testing Approaches

UI tests can be run manually or automatically.

©Simplilearn. All rights reserved.

Key Takeaways

- When an automated test fails, the TDD framework encourages developers to build new code.
- Git is simple to understand and use, with a small footprint and lightning-quick performance.
- Applications from many systems can operate simultaneously and in the same place due to Virtual Machines (VMs).
- The smallest part of the code can be logically separated into a system, often known as a unit.

©Simplilearn. All rights reserved.