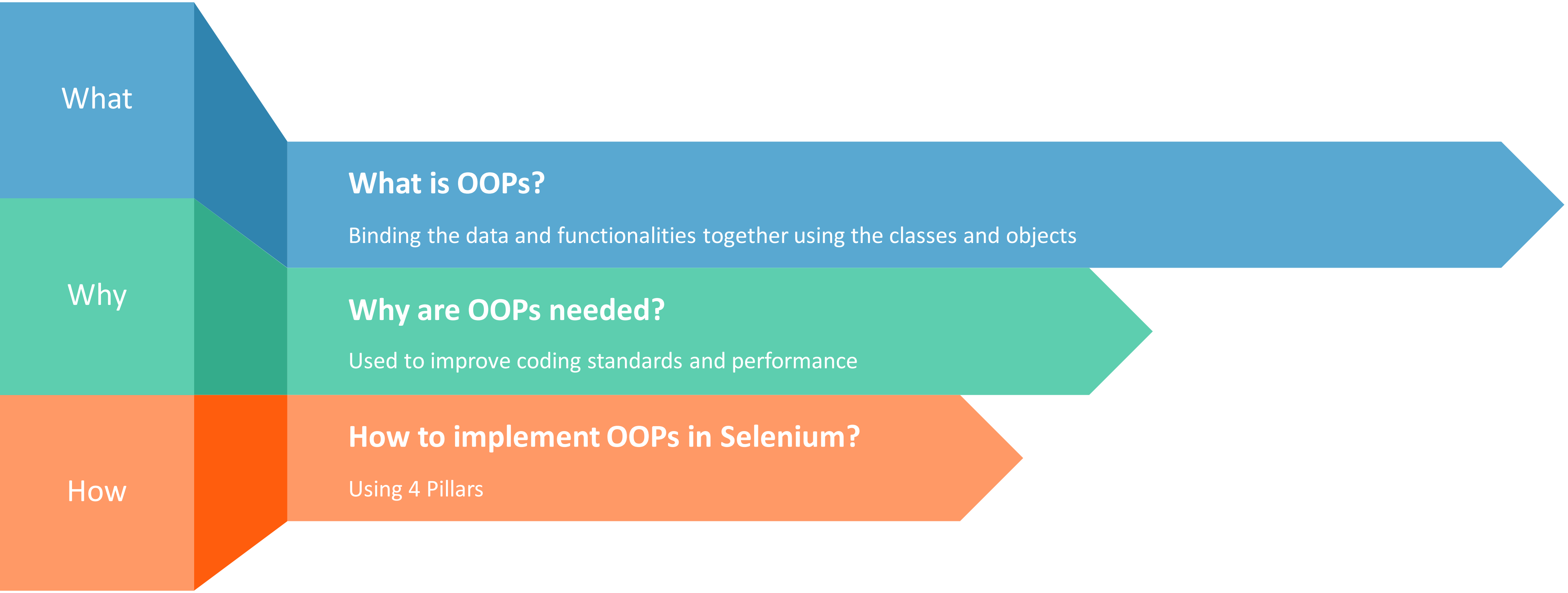
Selenium WebDriver Training

Java OOPs - Abstraction

Testleaf
Always Ahead

The Golden Circle



What

Why

How

What is OOPs?

Binding the data and functionalities together using the classes and objects

Why are OOPs needed?

Used to improve coding standards and performance

How to implement OOPs in Selenium?

Using 4 Pillars

Testleaf
Always Ahead

What is OOPs?

Programming standards

- that allows you to package together data states and functionality to modify those data states, while keeping the details hidden

Light Bulb – States?

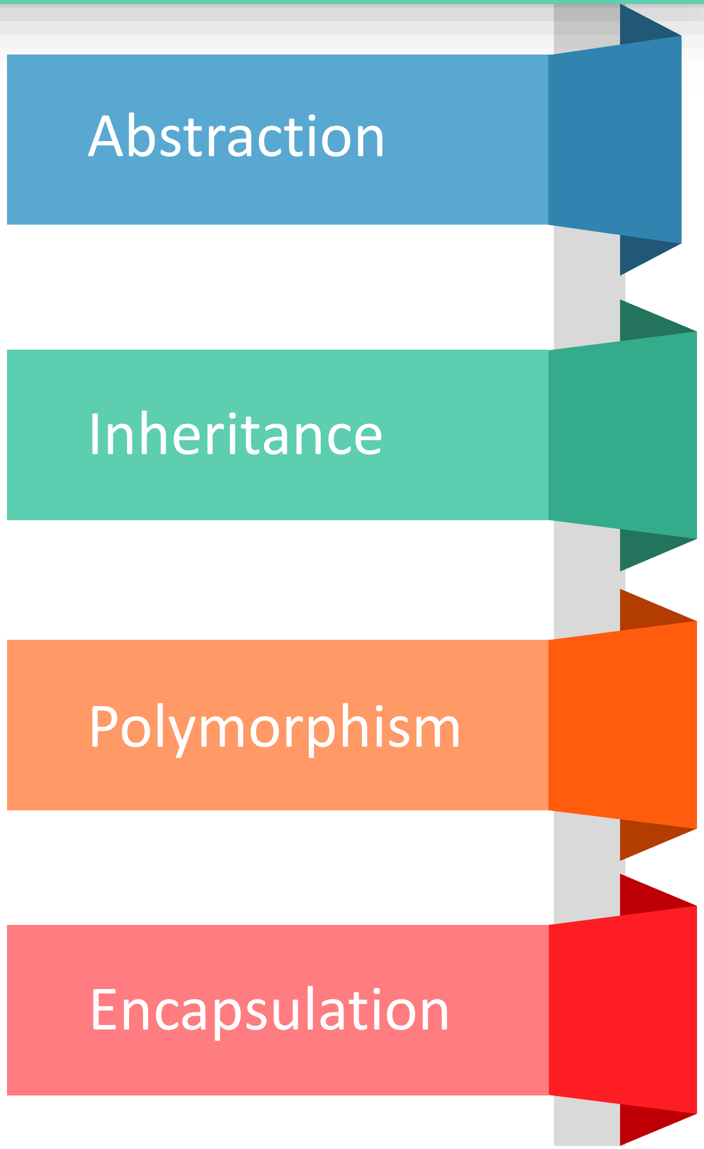
What is OOPs?

Programming standard

- that allows you to package together data states and functionality to modify those data states, while keeping the details hidden

Four Pillars of Object Orientation

- **Abstraction:** show the details that your user like to see and not the unimportant features.
- **Inheritance:** because a class can inherit attributes and behaviours from another class, you are able to reuse more code.
- **Polymorphism:** one class can be used to create many objects, all from the same flexible piece of code.
- **Encapsulation:** in OOP, you bundle code into a single unit where you can determine the scope of each piece of data.



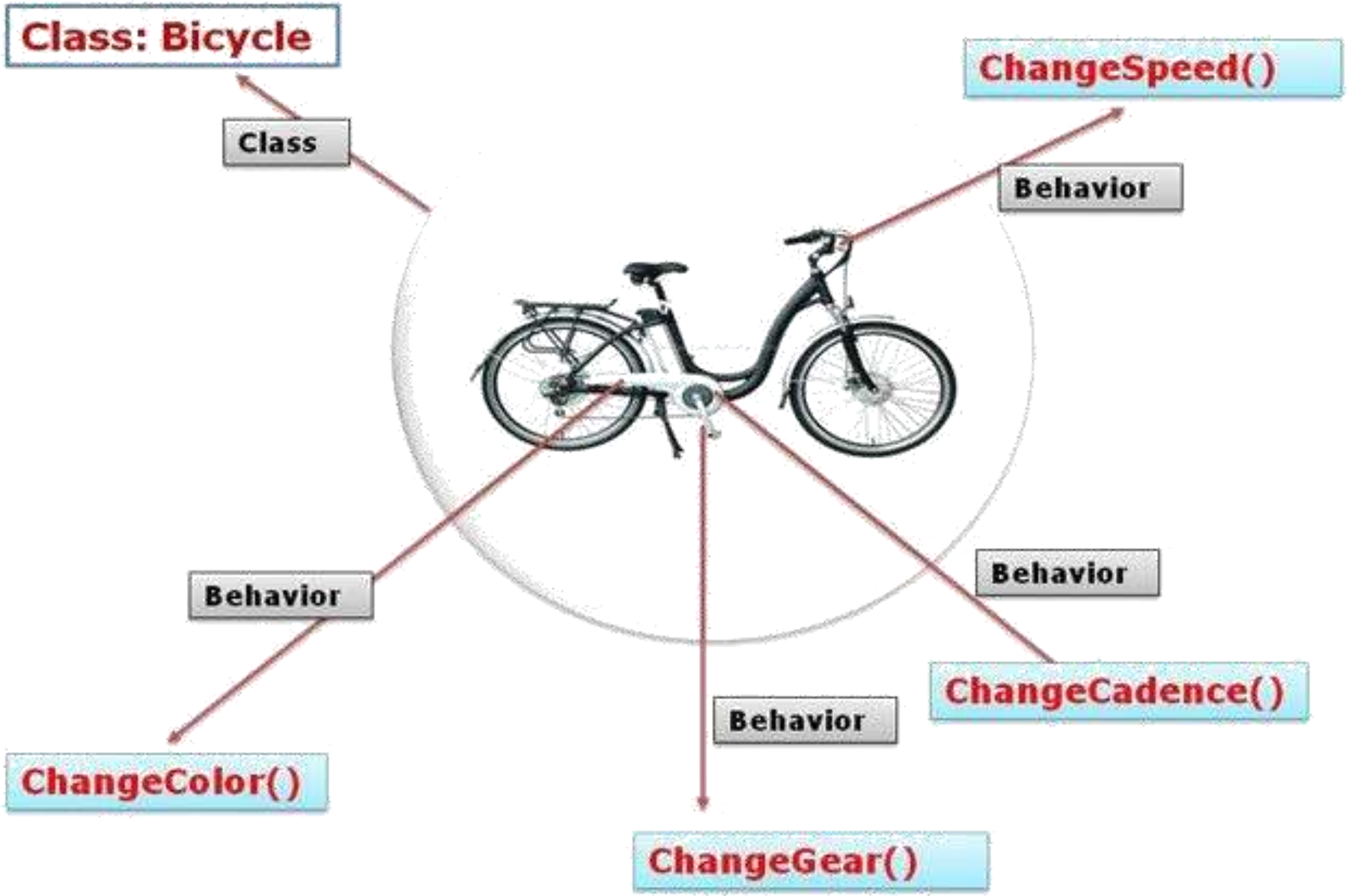
Abstraction

Inheritance

Polymorphism

Encapsulation

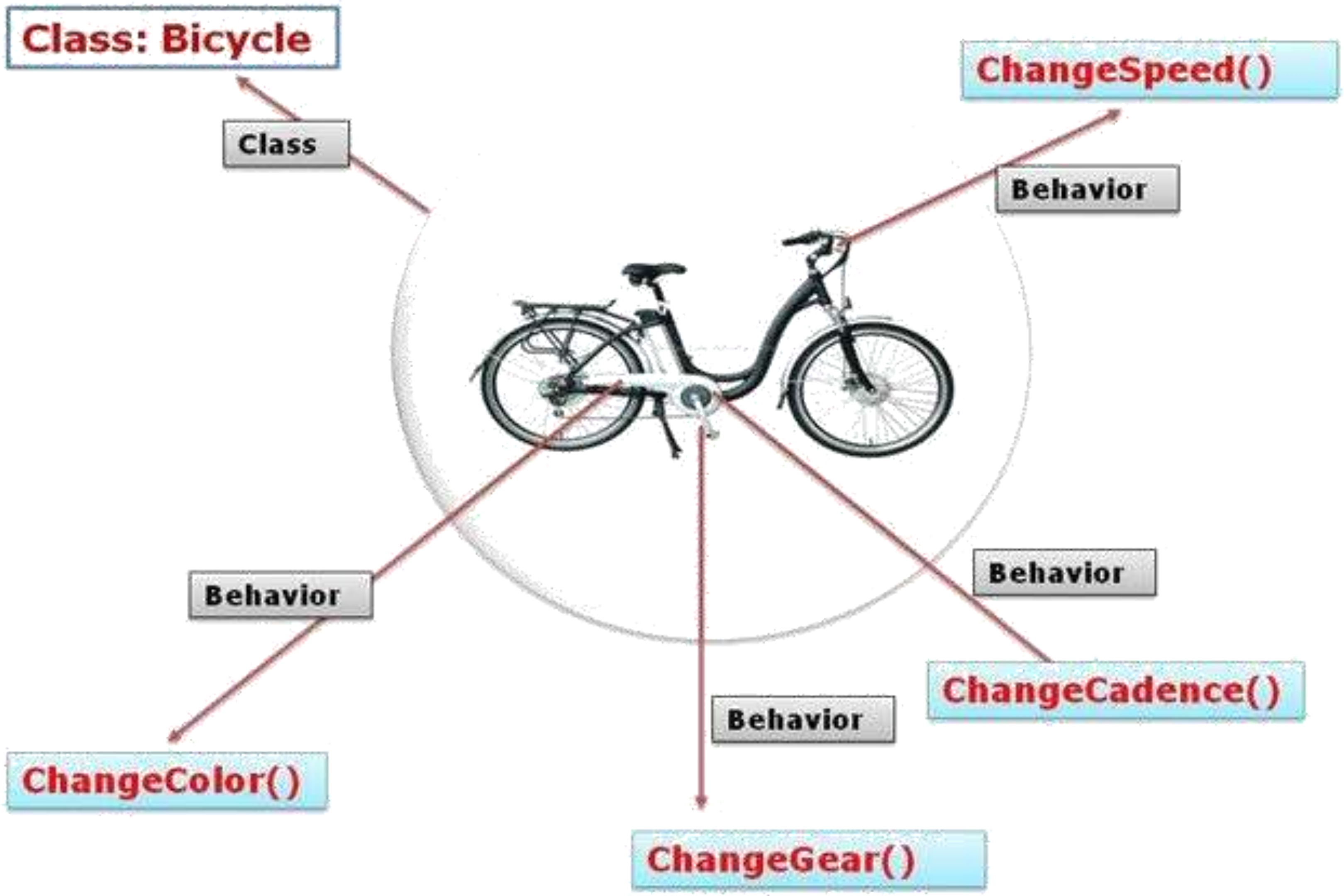
Class


Class: Bicycle
Class
ChangeSpeed()
Behavior
Behavior
ChangeColor()
Behavior
ChangeCadence()
Behavior
ChangeGear()

Object

Variable

Method


Class: Bicycle
Class
ChangeSpeed()
Behavior
Behavior
ChangeCadence()
Behavior
ChangeColor()
Behavior
ChangeGear()

Abstraction

- Display the details that your user like to see (bulb ON / OFF) and not the unimportant features.
- Can be achieved using interface and abstract class

Interface

- Blueprint of a class
- Compared to the architectural diagram of a house !

- 100% Abstract Methods (up to 1.7 version)

Banking !

ANDHRA BANK
AXIS BANK
BBK
Bank of Bahrain & Kuwait
Bank of Baroda
Bank of India
Bank of Maharashtra
केनरा बैंक
Canara Bank
citibank
CITY UNION BANK LTD
Corporation Bank
Deutsche Bank
FEDERAL BANK
HDFC BANK
ICICI Bank
IDBI BANK
Indian Overseas Bank
IndusInd Bank
ING
ING VYSYA BANK
J&K Bank
Karnataka Bank
Karur Vysya Bank
kotak
Kotak Mahindra Bank
LAKSHMI VILAS BANK
ORIENTAL BANK OF COMMERCE
punjab national bank

Banking !

- Bankers' Bank

RBI – Banker’ Bank

- Act as regulatory authority
- Any withdrawals / deposits above 10L should be reported !
- Mandatory KYC when you open account.

- Do RBI implement these methods? No

- Banks (SBI, ICICI, Axis ..) implement these !!!

Testleaf
Always Ahead

Interface

- Blueprint of a class
- Compared to the architectural diagram of a house !

- 100% Abstract Methods (up to 1.7 version)

Interface is implemented by the class ..

- Using keyword implements
- Syntax: Class implements Interface

- Should all the methods should be implemented? Yes, 100%

Interface is implemented by the class ..

- Using keyword implements
- Syntax: Class implements Interface

- If I do not implement, then? Compiler will not allow

Interface

- Blueprint of a class
- 100% Abstract Methods (up to 1.7 version)
- Uses implements keyword to make class to 100% implement
- We cannot create object for interface

Interface – RBI : Coding Example

- Let's create 3 abstract methods

Interface – RBI : Coding Example

- Let's create 3 abstract methods
- Create a concrete class (Axis) to implement them

Interface – RBI : Coding Example

- Let's create 3 abstract methods
- Create a concrete class (Axis) to implement them
- Another class (ICICI) to implement the same

Interface – RBI : Coding Example

- Let's create 3 abstract methods
- Create a concrete class (Axis) to implement them
- Another class (ICICI) to implement the same
- Add additional methods to the classes !!

Interface – RBI : Coding Example

- Let's create 3 abstract methods
- Create a concrete class (Axis) to implement them
- Another class (ICICI) to implement the same
- Add additional methods to the classes !!
- Check what happens when interface adds/delete methods

Interface – RBI : Coding Example

- Let's create 3 abstract methods
- Create a concrete class (Axis) to implement them
- Another class (ICICI) to implement the same
- Add additional methods to the classes !!
- Check what happens when interface adds/delete methods
- Check what happens when you create object for interface

Abstract Class

- A class that have both implementation and abstract.
- It can and cannot have abstract method (0 to 100%)
- We cannot create object for abstract class

- Syntax: abstract className

Summary

- Interfaces - 100% abstract
- Abstract Class – 0 to 100% abstract

Classroom Exercise (Breakout)

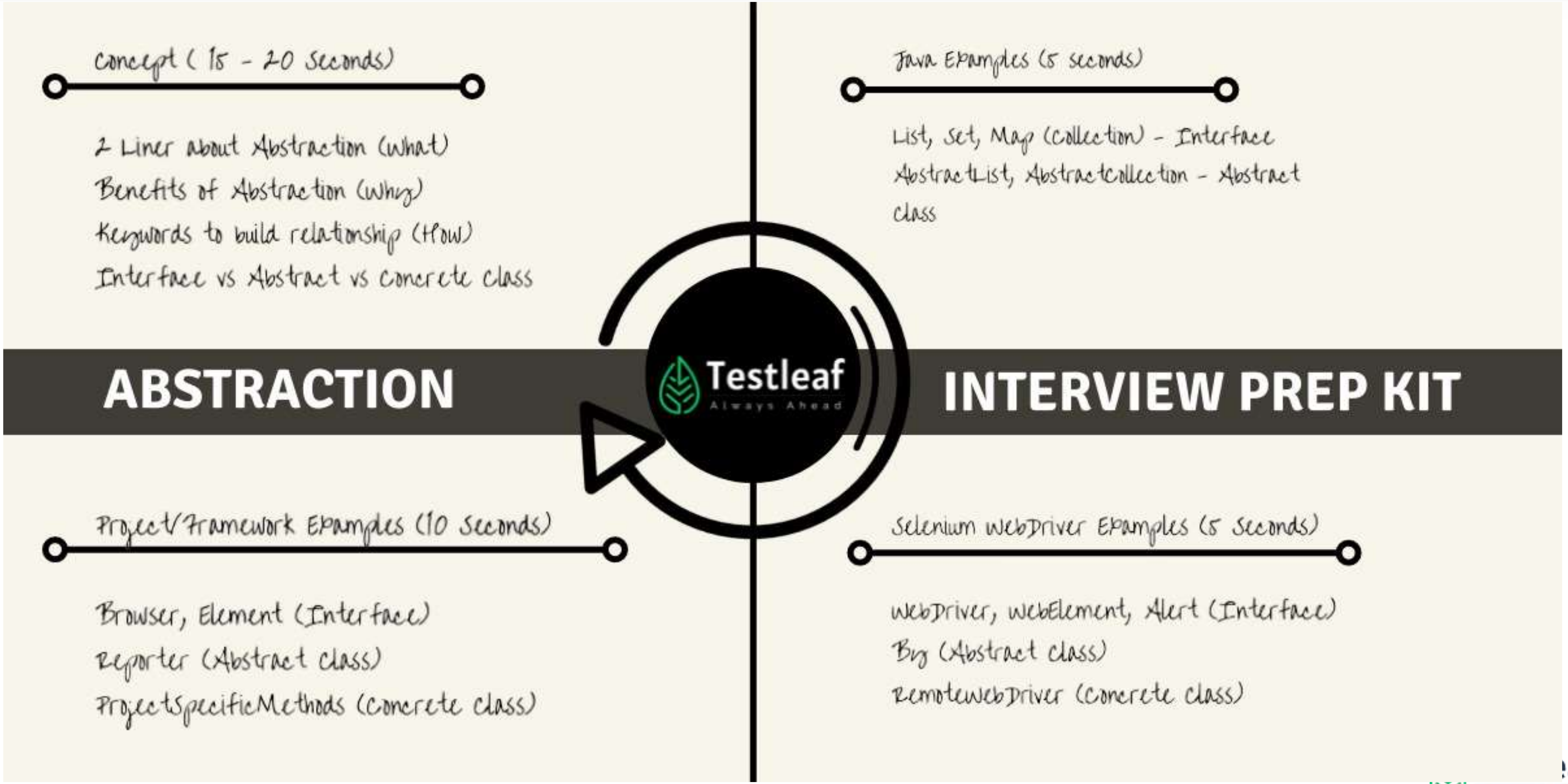
● Create an Interface for Android

- Open an App (openApp)
- Watch Video (playVideo)

● Implement using interface, abstract class

- Android (Interface)
- AndroidTV (Abstract class) – implement only openApp
- Samsung, OnePlus (Concrete Class) – implement the missing method

Interview Prep



ABSTRACTION

Testleaf

Always Ahead

INTERVIEW PREP KIT

Concept (15 - 20 Seconds)

2 Liner about Abstraction (what)
Benefits of Abstraction (why)
Keywords to build relationship (How)
Interface vs Abstract vs Concrete class

Java Examples (5 seconds)

List, Set, Map (Collection) - Interface
AbstractList, AbstractCollection - Abstract class

Project/Framework Examples (10 Seconds)

Browser, Element (Interface)
Reporter (Abstract class)
ProjectSpecificMethods (Concrete class)

Selenium WebDriver Examples (5 Seconds)

WebDriver, WebElement, Alert (Interface)
By (Abstract class)
RemoteWebDriver (Concrete class)