accenture

Strategy | Digital | Technology | Operations

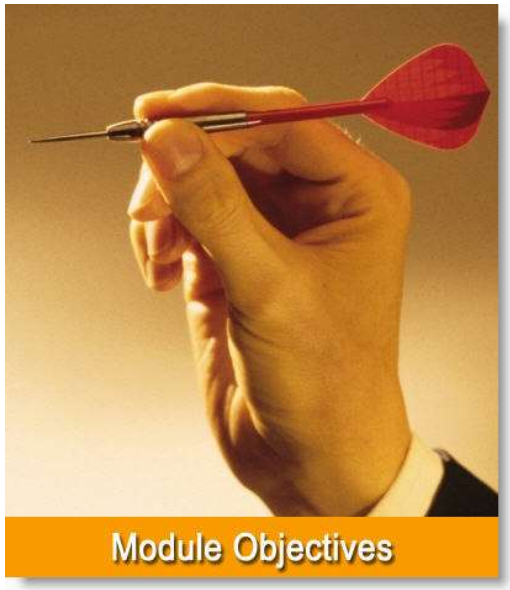
High performance. Delivered.

Advanced Programming: Java

Module 15 – Effective Test-Driven Development

Module Objectives

- No final deste módulo, os participantes serão capazes de:
  - Definir Extreme Programming (XP).
  - Explique o que são os testes de unidade e como eles funcionam.
  - Identifique quando e por que usar testes de unidade.
  - Explique a teoria do TDD.
  - Identifique os benefícios e as limitações do uso do TDD.
  - Aplique técnicas de desenvolvimento orientado a testes.



Agenda

- **XP Overview**
- Unit Testing
- Test Driven Development
- Benefits and Limitations of TDD

Extreme Programming (XP) Features

XP Features		
Extreme Programming	Programmer Centric	Test Centric
• É uma metodologia ágil que se concentra principalmente no lado da programação do desenvolvimento de software.	• É uma metodologia criada por programadores para programadores.	• É único entre as metodologias ágeis, devido ao seu foco semelhante ao laser nos testes.
• É leve, adapta-se a requisitos vagos e em rápida mudança, soluciona restrições de desenvolvimento de software e pode trabalhar com equipes de qualquer tamanho.		

XP Values

- XP é uma comunidade de prática de desenvolvimento de software baseada nos seguintes valores:

Communication

É bom conversar (principalmente entre usuários e desenvolvedore).

Simplicity

Mantenha-o simples e aumente o sistema conforme e quando necessário

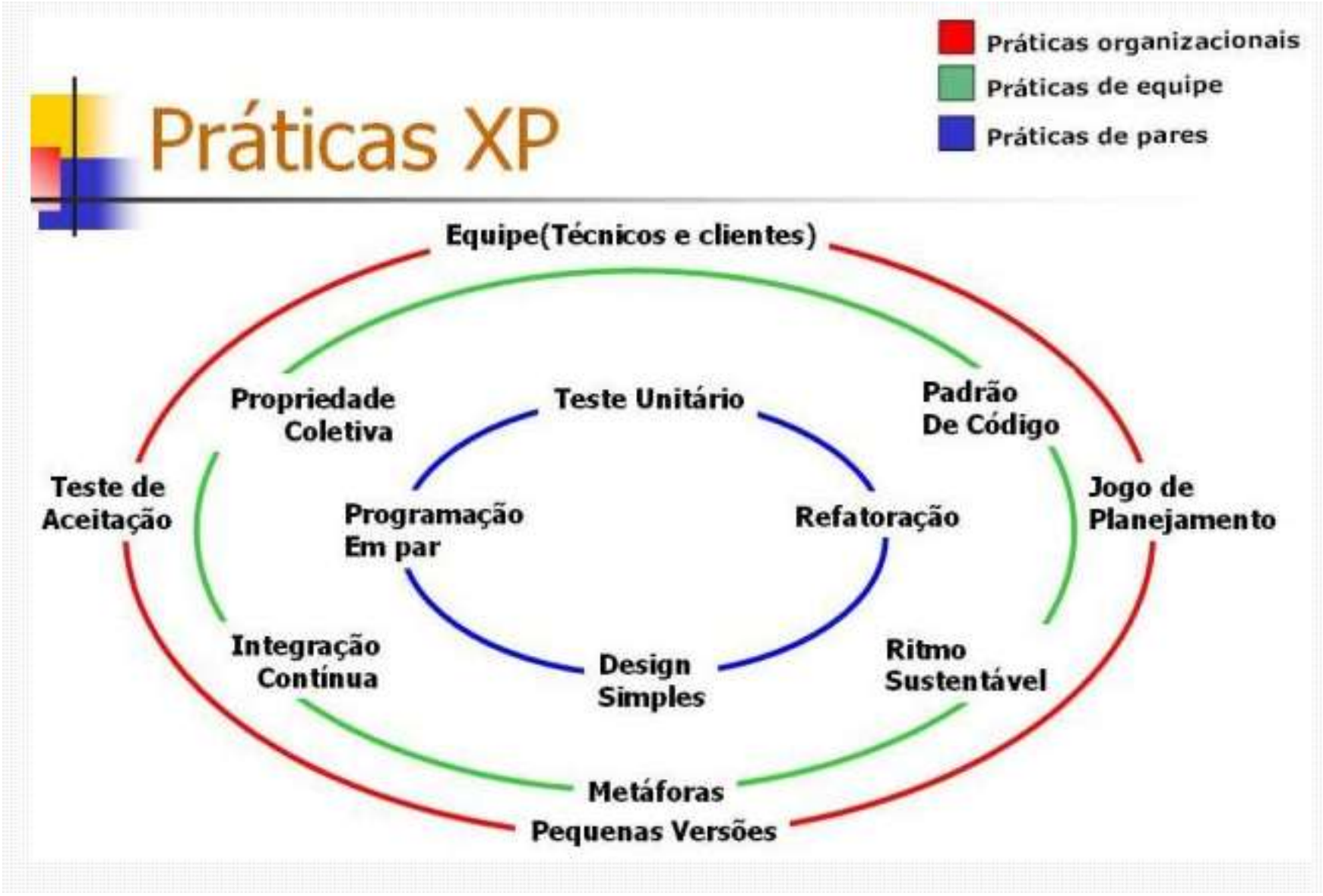
Feedback

Permita que os usuários forneçam feedback cedo e com frequência.

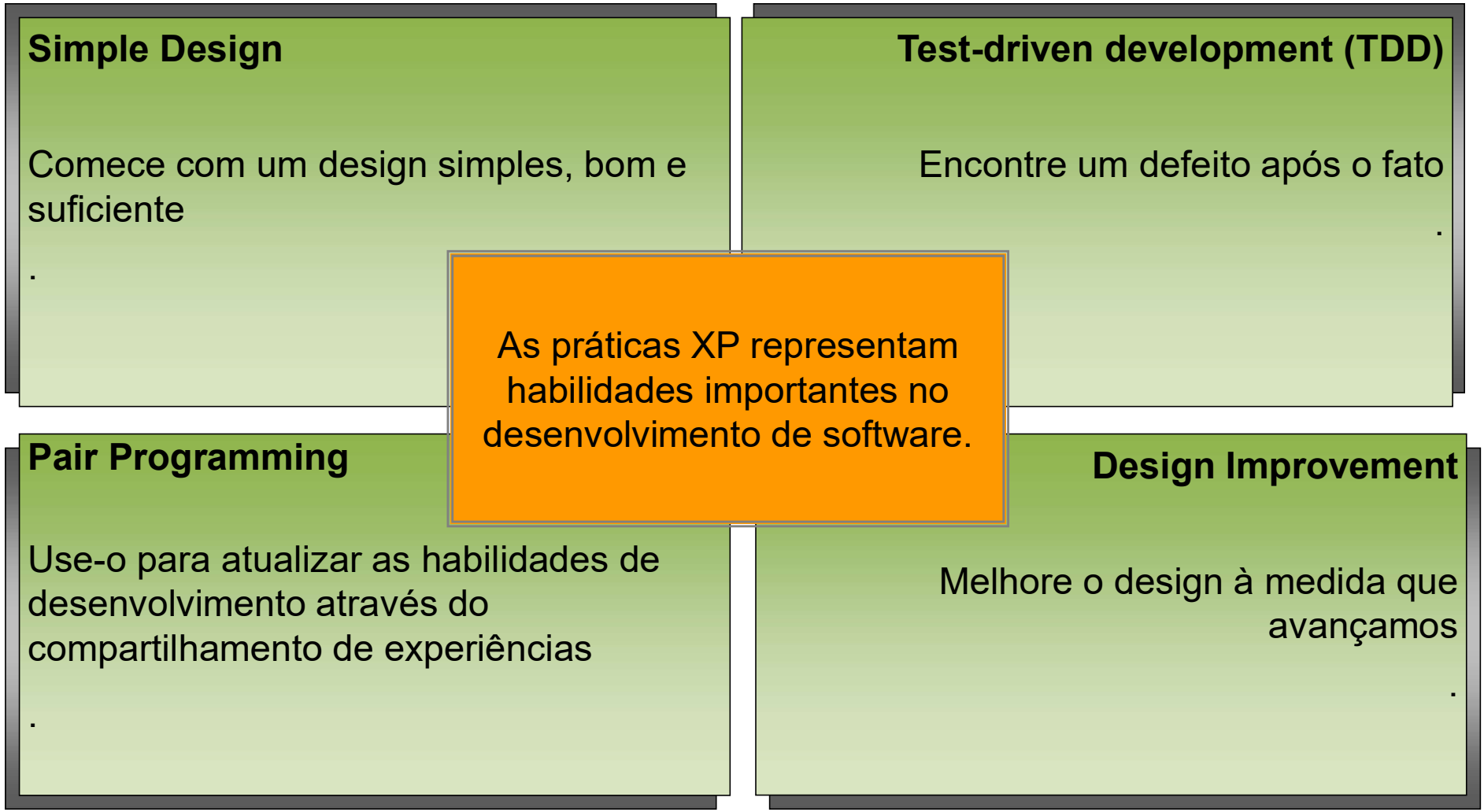
Courage

É preciso coragem para seguir essa abordagem.

Práticas XP



The Extreme Programming (XP) Practices



Agenda

- XP Overview
- **Unit Testing**
- Test Driven Development
- Benefits and Limitations of TDD

Definition

- O teste de unidade é uma maneira de testar partes individuais ou unidades de software conforme elas são criadas. Permitem a você:
  - Testar completamente todas as partes de uma unidade de código individual isoladamente.

- Os testes de unidade são escritos do ponto de vista do desenvolvedor:
  - São executadas para provar que um pedaço de código faz o que o desenvolvedor pensa que deve fazer.

Purpose

- O teste de unidade é a técnica mais eficaz para codificar melhor

- O objetivo do teste de unidade é isolar cada parte do programa e mostrar que as partes individuais estão corretas. Isso permitirá que você codifique com confiança.

- Os benefícios do teste de unidade incluem o seguinte::
  - Encontrar bugs mais cedo.
  - Criar uma melhor estrutura de software.
  - Melhorar o design.

Unit Test Frameworks

- O Teste de Unidade é a base do XP, que depende de uma estrutura de teste de unidade automatizada.
  - Várias estruturas de teste de unidade passaram a ser conhecidas coletivamente como xUnit.
  - As estruturas de teste de unidade são ferramentas de desenvolvimento.

- Eles foram desenvolvidos para uma ampla variedade de idiomas
  - Junit é o mais popular para linguagem Java;

- Esses frameworks são basicamente compostas por três objetos::
  - TestCase, TestSuite, and TestResult.

Java Unit Test Framework

- A estrutura de teste de unidade Java (JUnit) é a estrutura mais importante para escrever testes de unidade. Possui os seguintes recursos:
  - Fornece uma API.
  - Inclui ferramentas para executar seus testes e apresentar os resultados.
  - Permite que vários testes sejam agrupados para execução em lote.
  - É muito leve e simples de usar.
  - É projetado por desenvolvedores experientes para desenvolvedores experientes.
  - É extensível.

https://junit.org/junit5/docs/current/user-guide/#overview

JUnit: Java Unit Test Framework

- Para criar um teste de unidade, siga estas etapas básicas:

  1. Crie uma classe que estenda junit.framework.TestCase.

  2. Crie um método de void público dentro dessa classe cujo nome comece com "test".

  3. Nesse método, chame o código que deve ser testado.

```
import junit.framework.*;

public class TestSimple extends TestCase
{
   public TestSimple(String pName)
   {
      super(pName);
   }

   public void testAdd()
   {
      Simple vSimple = new Simple();
      assertEquals(2, vSimple.add(1,1));
   }
}

public class Simple
{
   public String add(int pFirst, int pSecond)
   {
      return pFirst + pSecond;
   }
}
```

Sonar

https://sonarcloud.io/explore/projects

Agenda

- XP Overview
- Unit Testing
- **Test Driven Development**
- Benefits and limitations of TDD

What is Test Driven Development (TDD)?

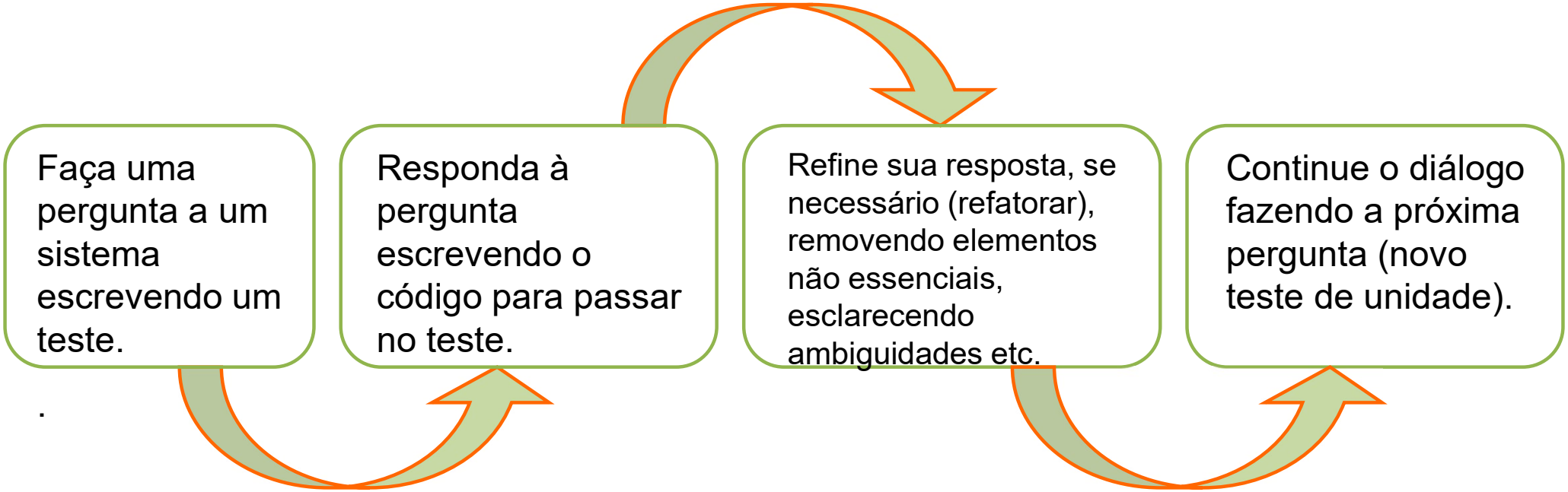
- Todos os desenvolvedores de software sabem que o teste é importante, mas quase ninguém o faz.
  - Design first, Test maybe.

- TDD é uma prática ágil para garantir que a qualidade seja construída desde o início.
  - Test first, Design always.

- O TDD baseia-se em dois conceitos principais:
  - Unit Tests
  - Refactoring

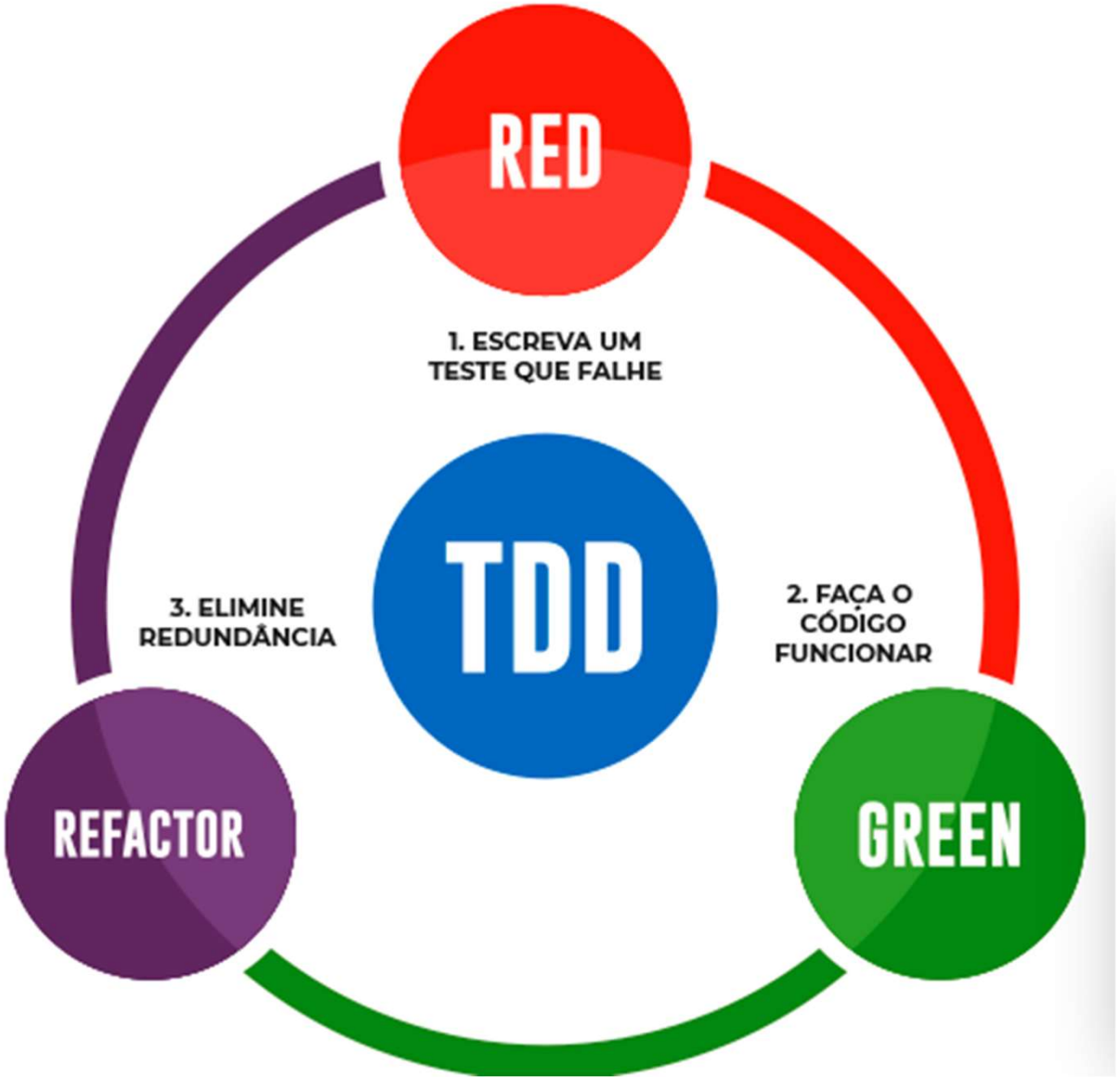
TDD Cycle

- O ciclo TDD transforma a programação em um diálogo.



.

TDD Cycle (cont.)



Mantra of TDD

- **Vermelho** (JUnit Fail Indication) ➔ **Verde** (JUnit Success Indication) ➔ ***Refactor*** (Melhore o design usando técnicas de refatoração) ➔ **Vermelho** ➔ **Verde** ➔ ***Refactor***➔…
  - As cores se referem ao que você vê quando escreve e executa um teste em uma ferramenta de teste de unidade (como JUnit). O processo é assim:
    - **Vermelho**: você cria um teste que expressa o que você espera que seu código fizesse. O teste falha (fica vermelho) porque você não criou código para fazer o teste passar.
    - **Verde**: você escreve um código para fazer o teste passar (ficar verde). Você não se preocupa em criar um design simples, claro e sem duplicação neste momento. Você seguirá para esse projeto mais tarde, quando o teste estiver passando e poderá experimentar confortavelmente projetos melhores. Se o design estiver correto, você pode ignorar a fase de refatoração e iniciar o ciclo com um novo teste.
    - Refatorador: Se necessário, você aprimora o design do código que passou no teste e passa para o próximo novo teste.

Agenda

- XP Overview
- Unit Testing
- Test Driven Development
- **Benefits and Limitations of TDD**

Discussion

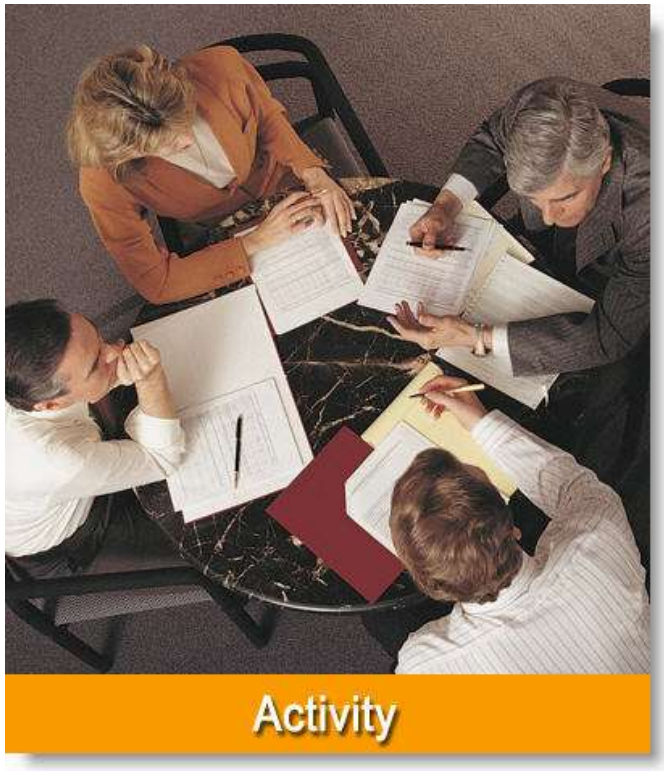
- Os testes podem tornar o processo de desenvolvimento mais produtivo?
- TDD é uma solução milagrosa?

Benefits and Limitations of TDD

Benefits	Limitations
• Desenvolvimento Simples e Incremental • Processo de desenvolvimento mais simples • Teste de regressão constante • Comunicação aprimorada • Compreensão aprimorada do comportamento necessário do software • Design de software aprimorado	• Custo inicial de tempo para : – A Curva de aprendizagem – Configuração

Activity

- Nesta atividade, você precisa atualizar a classe Employee para implementar as seguintes funções usando o TDD:
  - getFullName()
  - getTelephoneNumber()
  - payAmount().



Perguntas:

O que é extreme programming?

Extreme Programming (também conhecido como XP) é uma metodologia ágil e econômica para desenvolver softwares de alta qualidade com requisitos vagos e que se modificam a todo o momento.

O que é Unit Test?

O teste de unidade é uma maneira de testar partes individuais ou unidades de software conforme elas são construidas.

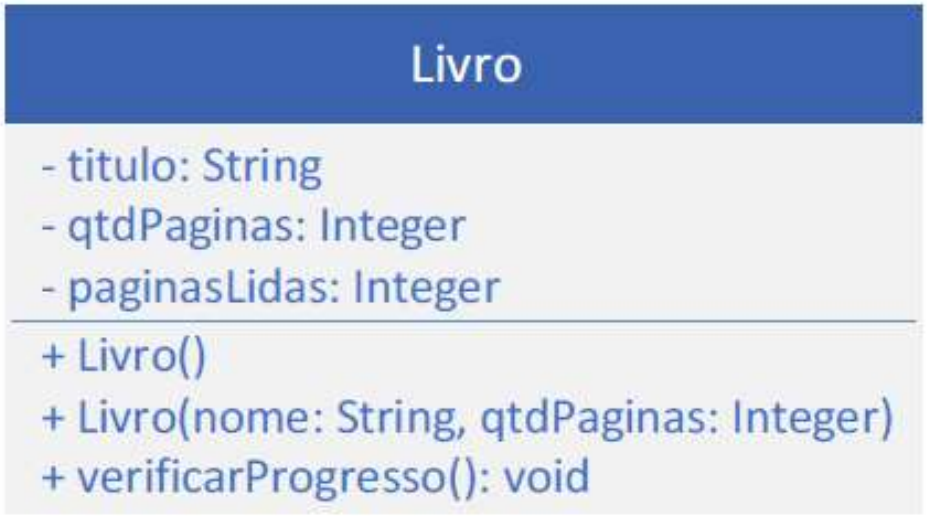
O que é TDD?

Prega que os testes devem ser criados antes da codificação;

Atividade

- **Atividade 20**

4) Construa a classe Livros em Java, que obedeça à descrição abaixo:



- A classe possui os atributos titulo, qtdPaginas e paginasLidas. Esses atributos devem ser marcados com o modificador de acesso private.
- Crie os métodos get e set para cada um dos atributos.
- Crie ainda o método verificarProgresso que deverá calcular a porcentagem de leitura do livro até o momento. Para isso, deverá usar os valores dos atributos qtdPaginas e paginasLidas, através da formula: porcentagem = paginasLidas * 100 / qtdPaginas. O valor da porcentagem deverá ser mostrado na tela conforme a mensagem "Você já leu X por cento do livro", onde o valor de X é o valor calculado pela fórmula apresentada anteriormente.

Atividade

- **Atividade 20**

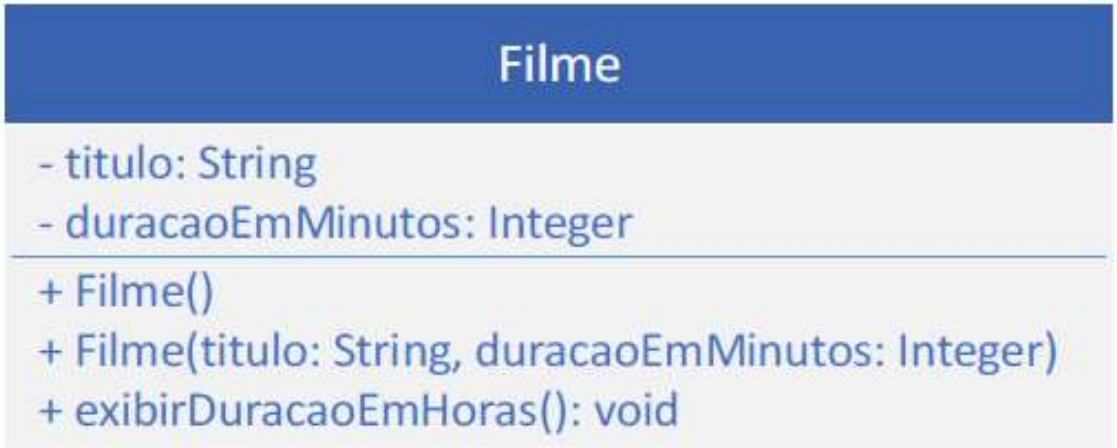
5) Crie uma classe **TestarLivros** no mesmo pacote da classe Livros da questão anterior. Essa classe possuirá apenas o método **main** que servirá para testar a classe Livros. As seguintes ações devem ser realizadas:

- Crie um objeto **livrofavorito** do tipo Livro.
- Altere o atributo **titulo** para "O Pequeno Príncipe". Utilize, para isso, o método **setTitulo**;
- Altere o atributo **qtdPaginas** para 98. Utilize, para isso, o método **setQtdPaginas**; Escreva na tela a mensagem: "O livro X possui Y páginas", onde no lugar de X deverá aparecer o valor do atributo **titulo** e, no lugar de Y deverá aparecer o valor do atributo **qtdPaginas**. Utilize, para tanto, os métodos **getTitulo** e **getQtdPaginas**.
- Altere a quantidade de **paginasLidas** para 20;
- Chame o método **verificarProgresso**.
- Altere a quantidade de paginasLidas para 50;
- Chame o método **verificarProgresso**.
- Instancie outros 10 livros no método main e chame os métodos desenvolvidos, conforme o exemplo anterior.

Atividade

- **Atividade 21**

6) Construa a classe Filmes em Java, que obedeça à descrição abaixo:



- A classe deve possuir dois atributos privados: titulo (do tipo String) e duracaoEmMinutos (do tipo int).
- Crie os métodos de acesso (get e set) para os atributos titulo e duracaoEmMinutos.
- Crie um método exibirDuracaoEmHoras que converta o valor em minutos para horas e apresente a mensagem "O filme TITULO possui X horas e Y minutos de duração".
- Por exemplo, dado o filme com título Titanic que possui 194 minutos de duração, a mensagem que deverá ser exibida para o usuário será:

"O filme Titanic possui 3 horas e 14 minutos de duração"

Atividade

- **Atividade 21**

7) Crie uma classe TestarFilme que possua um método main de modo que seja possível testar a classe Filme criada na questão anterior.

- Crie um objeto filme1 do tipo Filme.
- Altere o atributo título para “Os Vingadores”.
- Altere o atributo duracaoEmMinutos para 142.
- Chame o método exibirDuracaoEmHoras() para mostrar quantas horas o filme possui.
- Crie um objeto filme2 do tipo Filme.
- Altere o atributo título do filme2 para “Hotel Transilvânia”.
- Altere o atributo duracaoEmMinutos do filme2 para 93.
- Chame o método exibirDuracaoEmHoras() do filme2 para mostrar quantas horas o filme possui.
- Exiba a mensagem: “Os filmes em cartaz são X e Y”, onde no lugar de X, deverá aparecer o título do filme1 e no lugar de Y deverá aparecer o título do filme2.
- Instancie outros 5 filmes e faça as mesmas ações descritas acima, porém utilizando novos valores.

Questions and Comments

- What questions or comments do you have?