

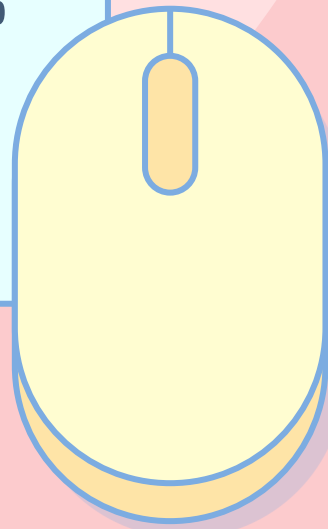


JUnit

Gabriel Faria, Maisa Pires e Miguel Vieira

Enter

Sobre o JUnit

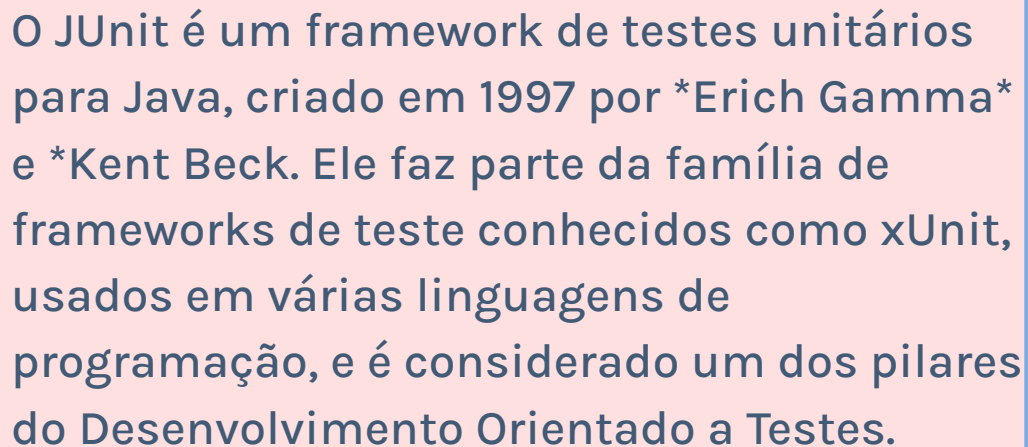




A



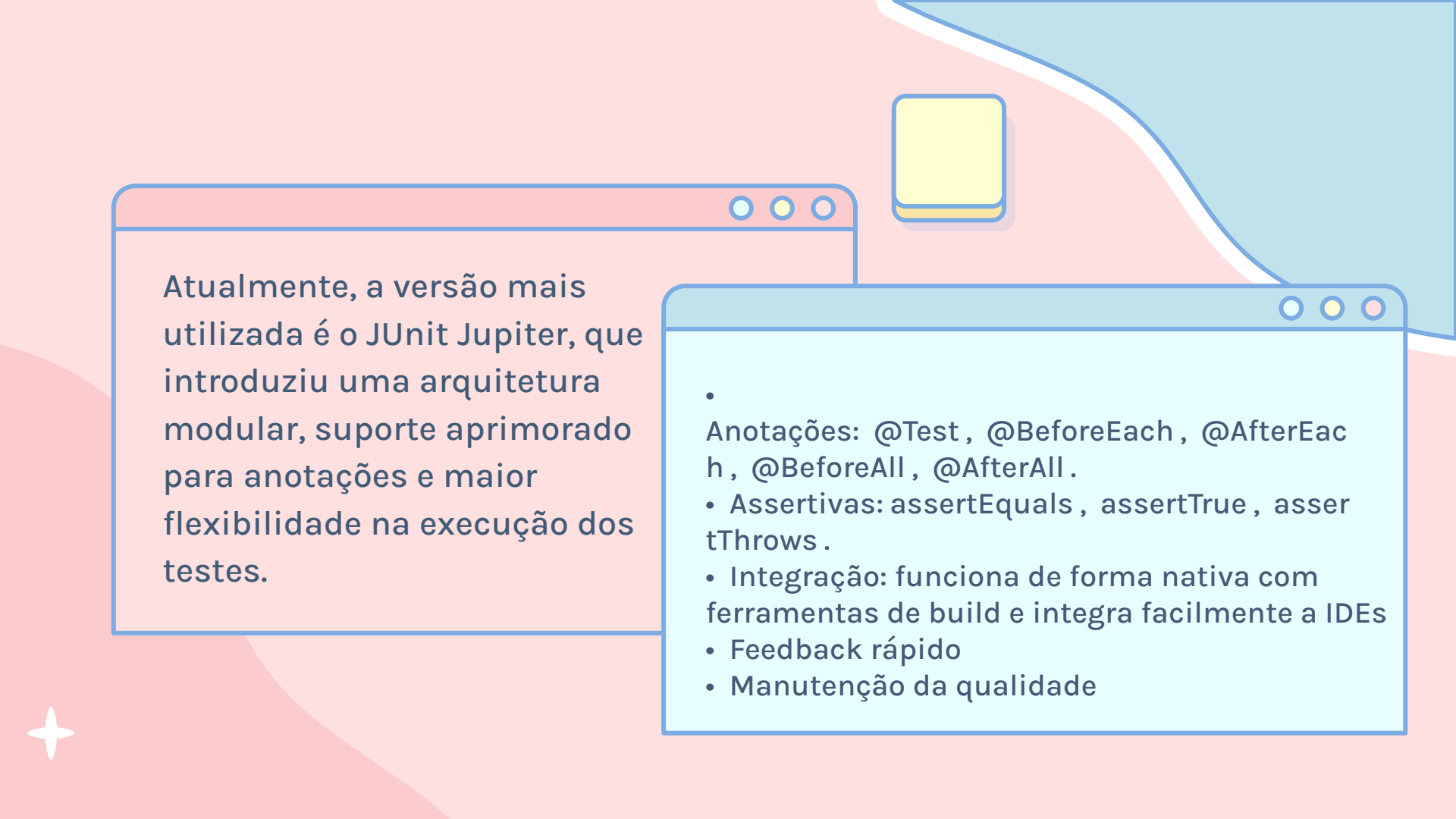
Cmd



O JUnit é um framework de testes unitários para Java, criado em 1997 por *Erich Gamma* e *Kent Beck. Ele faz parte da família de frameworks de teste conhecidos como xUnit, usados em várias linguagens de programação, e é considerado um dos pilares do Desenvolvimento Orientado a Testes.



Ctrl

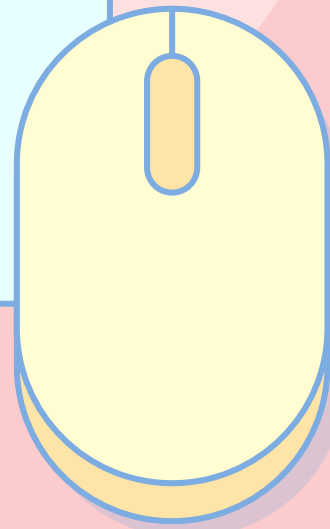


Atualmente, a versão mais utilizada é o JUnit Jupiter, que introduziu uma arquitetura modular, suporte aprimorado para anotações e maior flexibilidade na execução dos testes.

- Anotações: `@Test`, `@BeforeEach`, `@AfterEach`, `@BeforeAll`, `@AfterAll`.
- Assertivas: `assertEquals`, `assertTrue`, `assertThrows`.
- Integração: funciona de forma nativa com ferramentas de build e integra facilmente a IDEs
- Feedback rápido
- Manutenção da qualidade

Enter

Categorização do Framework



Técnicas de Teste

O JUnit se encaixa principalmente em testes de caixa branca

Níveis de Teste

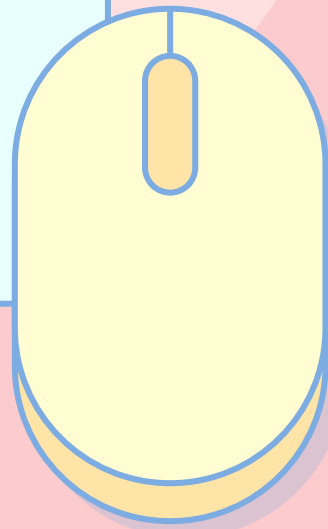
É mais utilizado em teste de unidade, validando métodos e classes individualmente. No entanto, também pode ser estendido para testes de integração, especialmente quando combinado com bibliotecas de simulação

Tipos de Teste

- Testes funcionais: garante que os métodos realizem corretamente suas operações.
- Testes de regressão: sempre que o código muda, os testes podem ser reexecutados para verificar se nada foi quebrado.
- Testes automatizados: facilitando a repetição sem intervenção manual.

Enter

Instalação/ Integração



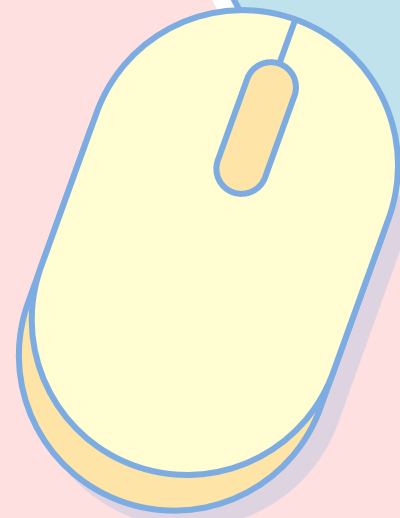
Maven

```
<dependency>  
<groupId>org.junit.jupiter</groupId>  
<artifactId>junit-jupiter</artifactId>  
<version>5.10.0</version>  
<scope>test</scope>  
</dependency>
```

Gradle

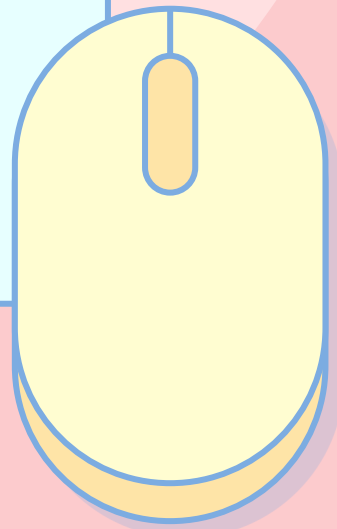
```
testImplementation  
'org.junit.jupiter:junit-jup  
iter:5.10.0'
```

Em IDEs modernas, basta
criar o projeto Java com
suporte a testes e
adicionar a dependência
para começar.



Estratégia e casos de testes


Enter




Foi aplicada a estratégia de Particionamento de Equivalência, consideramos valores-limites como zero e números negativos


Casos de testes

- Soma de dois números positivos.
- Soma envolvendo número zero.
- Soma de números negativos.
- Subtração de dois números positivos.
- Subtração envolvendo número negativo.
- Multiplicação de dois números positivos.
- Multiplicação por zero.
- Multiplicação envolvendo número negativo.



```
public class Calculadora {  
  
    public int somar(int a, int b) {  
        return a + b;  
    }  
  
    public int subtrair(int a, int b) {  
        return a - b;  
    }  
  
    public int multiplicar(int a, int b) {  
        return a * b;  
    }  
}
```







```
class CalculadoraTest {

    @Test
    void somarPositivos() {
        Calculadora calc = new Calculadora();
        assertEquals(expected:5, calc.somar(a:2, b:3));
    }

    @Test
    void somarComZero() {
        Calculadora calc = new Calculadora();
        assertEquals(expected:7, calc.somar(a:7, b:0));
    }


    @Test
    void somarNegativos() {
        Calculadora calc = new Calculadora();
        assertEquals(-5, calc.somar(-2, -3));
    }
}
```






```
@Test
void subtrairPositivos() {
    Calculadora calc = new Calculadora();
    assertEquals(expected:1, calc.subtrair(a:3, b:2));
}
```

```
@Test
void subtrairComZero() {
    Calculadora calc = new Calculadora();
    assertEquals(expected:0, calc.subtrair(a:2, b:0));
}
```




```
@Test
void subtrairComNegativo() {
    Calculadora calc = new Calculadora();
    assertEquals(expected:5, calc.subtrair(a:2, -3));
}
```



```
@Test
void multiplicarPositivos() {
    Calculadora calc = new Calculadora();
    assertEquals(expected:6, calc.multiplicar(a:2, b:3));
}
```

```
@Test
void multiplicarPorZero() {
    Calculadora calc = new Calculadora();
    assertEquals(expected:0, calc.multiplicar(a:5, b:0));
}
```



```
@Test
void multiplicarComNegativo() {
    Calculadora calc = new Calculadora();
    assertEquals(-10, calc.multiplicar(-2, b:5));
}
}
```

Cmd

+

A

Obrigado!

Ctrl

Z

C

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**

Please keep this slide for attribution

Referências

- <https://www.devmedia.com.br/junit-tutorial/1432>
- <https://github.com/junit-team/junit-framework>