Fernanda Pinheiro Reis
1110481823022 ADS - Noite - Testes de Software

---

1. Cria o script de teste REQ01CadastrarAluno. Organize o código de maneira a permitir a inspeção por pares (no Word o espaçamento entre linhas deve ser simples e o espaçamento antes da linha e depois da linha deve ser 0).

1.1. Criar o caso de teste – cadastrar aluno com sucesso

```
@Test
	void cadastro_com_sucesso() {
		repository.deleteAll();
		repository.save(new Aluno(1L, "XXXXX", "Cris",
"cris@fatec.sp.gov.br", "21312354", "Rua Sete, 7"));
		assertEquals(1, repository.count());
	}
```

1.2. Criar o caso de teste – para validação dos dados de entrada

```
@Test
	void validacao_de_entrada_com_sucesso() {
		repository.deleteAll();
		validatorFactory =
Validation.buildDefaultValidatorFactory();
		validator = validatorFactory.getValidator();
		Aluno aluno = new Aluno(1L, "XXXXX", "Cris",
"cris@fatec.sp.gov.br", "21312354", "Rua Sete, 7");
		Set<ConstraintViolation<Aluno>> violations =
validator.validate(aluno);
		assertTrue(violations.isEmpty());
	}
```

1.3. Criar o caso de teste – para validação de dados de entrada inválidos – atributo nome do aluno em branco

```
@Test
	void validacao_de_entrada_dados_invalidos() {
		validatorFactory =
Validation.buildDefaultValidatorFactory();
		validator = validatorFactory.getValidator();
		repository.deleteAll();
		Aluno aluno = new Aluno(1L, "XX", "Cris",
"cris@fatec.sp.gov.br", "21312354", "Rua Sete, 7");
		Set<ConstraintViolation<Aluno>> violations =
validator.validate(aluno);
		assertEquals("O RA deve conter entre 3 e 10
caracteres.", violations.iterator().next().getMessage());
	}
```

1.4. Criar o caso de teste – para validação de RA já cadastrado

```
@Test
	void validacao_de_RA_cadastrado() {
		repository.deleteAll();
		Aluno aluno = new Aluno(1L, "XXXXX", "Juce",
"juce@fatec.sp.gov.br", "21312354", "Rua das Vilas, 66");
		Aluno alunoRepetido = new Aluno(1L, "XXXXX",
"Juce", "juce@fatec.sp.gov.br", "21312354", "Rua das Vilas, 66");
		repository.save(aluno);
		try {
			repository.save(alunoRepetido);
		} catch (Exception e) {
			System.out.println(e);
		}
		assertEquals(1, repository.count());
	}
```

2. Criar o script de teste REQ02ConsultarAluno.

2.1. Criar o caso de teste que valida a consulta com sucesso

```
@Test
	void validacao_consulta_sucesso() {
		repository.deleteAll();
		Aluno aluno = new Aluno(1L, "XXXXX", "Cris",
"cris@fatec.sp.gov.br", "21312354", "Rua Sete, 7");
		repository.save(aluno);
		Optional<Aluno> alunoConsulta =
repository.findByRa("XXXXX");

assertThat(alunoConsulta.get().getRa()).isEqualTo(aluno.getR
a());
	}
```

2.2. Cria o caso de teste que valida a consulta com nome parcial

```
@Test
	void validacao_consulta_nome_parcial() {
		repository.deleteAll();
		Aluno aluno1 = new Aluno(1L, "XXXXX", "Cris",
"cris@fatec.sp.gov.br", "21312354", "Rua Sete, 7");
		repository.save(aluno1);
		Aluno aluno2 = new Aluno(null, "ZZZZZ", "Clau",
"clau@fatec.sp.gov.br", "67564566", "Rua da Felicidade, 9");
		repository.save(aluno2);
		Aluno aluno3 = new Aluno(null, "YYYYY", "Binn",
"binn@fatec.sp.gov.br", "20202020", "Rua do ano sem fim,
2020");
		repository.save(aluno3);
		List<Aluno> alunos =
repository.findAllByNomeIgnoreCaseContaining("Cris");
		assertThat(alunos.size()).isEqualTo(1);
	}
```

3. Criar o script de teste REQ03ExcluirAluno. Organize o código de maneira a permitir a inspeção por pares

3.1. Criar um caso de teste que valida a exclusão de um aluno do cadastro

```
@Test
	void validacao_exclusao() {
		repository.deleteAll();
		Aluno aluno = new Aluno(1L, "XXXXX", "Cris",
"cris@fatec.sp.gov.br", "21312354", "Rua Sete, 7");
		repository.save(aluno);
		Optional<Aluno> alunoFind =
repository.findByRa("XXXXX");
		repository.deleteById(alunoFind.get().getId());

assertThat(repository.findByRa("XXXXX")).isEmpty();
	}
```