В Jest и Testing Library React используются матчеры для проверки различных условий в тестах. Эти матчеры делают тесты выразительными и читаемыми.

**Матчеры Jest**

**1. Матчеры для основных операций**

toBe(value)

  Проверяет, что значение строго равно (===) указанному.

    expect(2 + 2).toBe(4);

 toEqual(value)

  Проверяет глубокое равенство объектов или массивов.

    expect({ foo: 'bar' }).toEqual({ foo: 'bar' });

 not

  Обратная проверка (отрицание).

    expect(2 + 2).not.toBe(5);

**2. Проверка чисел**

toBeGreaterThan(number)\*\*

  Проверяет, что значение больше указанного.

    expect(10).toBeGreaterThan(5);

 toBeGreaterThanOrEqual(number)\*\*

  Проверяет, что значение больше или равно указанному.

    expect(10).toBeGreaterThanOrEqual(10);

 toBeLessThan(number)\*\*

  Проверяет, что значение меньше указанного.

    expect(5).toBeLessThan(10);

 toBeLessThanOrEqual(number)\*\*

  Проверяет, что значение меньше или равно указанному.

    expect(5).toBeLessThanOrEqual(5);

 toBeCloseTo(number, numDigits?)\*\*

  Проверяет равенство чисел с учетом округления (полезно для вычислений с плавающей точкой).

    expect(0.1 + 0.2).toBeCloseTo(0.3);

**3. Работа со строками**

toMatch(regexpOrString)\*\*

  Проверяет, что строка соответствует регулярному выражению или включает подстроку.

  expect('Hello, World!').toMatch(/world/i);

  expect('Hello, Jest!').toMatch('Jest');

**4. Работа с массивами и объектами**

toContain(item)\*\*

  Проверяет наличие элемента в массиве или строки в Set.

    expect(['a', 'b', 'c']).toContain('b');

  expect(new Set(['abc', 'def'])).toContain('abc');

 toContainEqual(item)\*\*

  Проверяет наличие глубоко равного объекта в массиве.

    expect([{ foo: 'bar' }]).toContainEqual({ foo: 'bar' });

 toHaveLength(number)\*\*

  Проверяет длину массива или строки.

    expect([1, 2, 3]).toHaveLength(3);

 toHaveProperty(propertyPath, value?)\*\*

  Проверяет наличие свойства у объекта (и значение, если указано).

    expect({ user: { name: 'John' } }).toHaveProperty('user.name', 'John');

**5. Работа с функциями**

toHaveBeenCalled()\*\*

  Проверяет, что mock-функция была вызвана.

    const mockFn = jest.fn();

  mockFn();

  expect(mockFn).toHaveBeenCalled();

 toHaveBeenCalledTimes(number)\*\*

  Проверяет, сколько раз функция была вызвана.

    expect(mockFn).toHaveBeenCalledTimes(1);

 toHaveBeenCalledWith(...args)\*\*

  Проверяет, что функция была вызвана с указанными аргументами.

    mockFn(1, 2, 3);

  expect(mockFn).toHaveBeenCalledWith(1, 2, 3);

 toHaveReturnedWith(value)\*\*

  Проверяет, что функция вернула указанное значение.

    const mockFn = jest.fn(() => 42);

  expect(mockFn).toHaveReturnedWith(42);

**6. Работа с промисами**

resolves (для асинхронных функций/промисов)

  Проверяет успешную обработку промиса.

    await expect(Promise.resolve(42)).resolves.toBe(42);

 rejects (для асинхронных функций/промисов)

  Проверяет отклонение промиса.

    await expect(Promise.reject(new Error('fail'))).rejects.toThrow('fail');

**7. Работа с ошибками**

toThrow(error?)\*\*

  Проверяет, что функция выбросила ошибку.

    const throwError = () => { throw new Error('ошибка'); };

  expect(throwError).toThrow('ошибка');

**Матчеры Testing Library React (jest-dom)**

Testing Library предоставляет дополнительные матчеры через библиотеку @testing-library/jest-dom, часто используемую для тестирования DOM-структуры.

**1. Проверка наличия в DOM**

toBeInTheDocument()\*\*

  Проверяет, что элемент находится в DOM.

    const { getByText } = render(Привет);

  expect(getByText(/привет/i)).toBeInTheDocument();

 toBeVisible()\*\*

  Проверяет, что элемент виден для пользователя (не скрыт стилями).

    expect(buttonElement).toBeVisible();

**2. Проверка атрибутов**

toHaveAttribute(name, value?)\*\*

  Проверяет наличие атрибута и его значения.

    expect(linkElement).toHaveAttribute('href', '/about');

 toHaveClass(...classNames)\*\*

  Проверяет наличие классов у элемента.

    expect(button).toHaveClass('btn-primary');

 toHaveStyle(cssStringOrObject)\*\*

  Проверяет наличие или соответствие стилей.

    expect(div).toHaveStyle('display: none');

**3. Проверка текста и содержимого**

toHaveTextContent(text, options?)\*\*

  Проверяет, что элемент содержит указанный текст.

    expect(element).toHaveTextContent('Привет, мир!');

 toContainElement(element)\*\*

  Проверяет, что один элемент содержит другой элемент.

    expect(parentElement).toContainElement(childElement);

**4. Проверка форм и взаимодействий**

toBeChecked()\*\*

  Проверяет, что ` типа checkbox или radio` установлен в состояние checked.

    expect(checkbox).toBeChecked();

 toBeDisabled() / toBeEnabled()\*\*

  Проверяет, что элемент в состоянии disabled или enabled.

    expect(button).toBeDisabled();

 toHaveValue(value)\*\*

  Проверяет значение элементов ввода (`, ` и др.).

    expect(input).toHaveValue('test');

**5. Проверка фокуса**

**toHaveFocus()\*\***

  Проверяет, что элемент имеет фокус.

    expect(inputElement).toHaveFocus();

});