import {

$,

swal,

removeSwal,

$$,

onAction,

CLASS\_NAMES,

delay,

} from './utils';

const {

BUTTON,

CONFIRM\_BUTTON,

CANCEL\_BUTTON,

MODAL,

} = CLASS\_NAMES;

afterEach(() => removeSwal());

describe("show buttons", () => {

test("shows only confirm button by default", () => {

swal();

expect($$(BUTTON).length).toBe(1);

expect($$(BUTTON).hasClass(CONFIRM\_BUTTON)).toBeTruthy();

});

test("hides all buttons", () => {

swal({

buttons: false,

});

expect($$(BUTTON).length).toBe(0);

});

test("shows confirm and cancel buttons", () => {

swal({

buttons: true,

});

expect($$(BUTTON).length).toBe(2);

expect($$(CONFIRM\_BUTTON).length).toBe(1);

expect($$(CANCEL\_BUTTON).length).toBe(1);

});

test("sets button text", () => {

swal({

button: "Test",

});

expect($$(CONFIRM\_BUTTON).text()).toBe("Test");

});

test("sets button texts with array", () => {

swal({

buttons: ["Stop", "Do it"],

});

expect($$(CONFIRM\_BUTTON).text()).toBe("Do it");

expect($$(CANCEL\_BUTTON).text()).toBe("Stop");

});

test("sets default button texts with array", () => {

swal({

buttons: [true, true],

});

expect($$(CONFIRM\_BUTTON).text()).toBe("OK");

expect($$(CANCEL\_BUTTON).text()).toBe("Cancel");

});

test("uses button object", () => {

swal({

buttons: {

cancel: "Run away!",

confirm: true,

}

});

expect($$(CANCEL\_BUTTON).text()).toBe("Run away!");

expect($$(CONFIRM\_BUTTON).text()).toBe("OK");

});

test("sets more than 2 buttons", () => {

swal({

buttons: {

cancel: "Run away!",

catch: {

text: "Throw Pokéball!",

},

defeat: true,

},

});

expect($$(BUTTON).length).toBe(3);

expect($$(CANCEL\_BUTTON).text()).toBe("Run away!");

expect($$(CONFIRM\_BUTTON).length).toBe(0);

expect($$(`${BUTTON}--catch`).length).toBe(1);

expect($$(`${BUTTON}--catch`).text()).toBe("Throw Pokéball!");

expect($$(`${BUTTON}--defeat`).length).toBe(1);

expect($$(`${BUTTON}--defeat`).text()).toBe("Defeat");

});

});

describe("buttons resolve values", () => {

test("confirm button resolves to true", async () => {

expect.assertions(1);

setTimeout(() => {

$$(CONFIRM\_BUTTON).click();

}, 500);

const value = await swal();

expect(value).toBeTruthy();

});

test("cancel button resolves to null", async () => {

expect.assertions(1);

setTimeout(() => {

$$(CANCEL\_BUTTON).click();

}, 500);

const value = await swal({

buttons: true,

});

expect(value).toBeNull();

});

test("can specify resolve value", async () => {

expect.assertions(1);

setTimeout(() => {

$$(CONFIRM\_BUTTON).click();

}, 500);

const value = await swal({

button: {

value: "test",

},

});

expect(value).toBe("test");

});

test("extra button resolves to string by default", async () => {

expect.assertions(1);

setTimeout(() => {

$(`.${BUTTON}--test`).click();

}, 500);

const value = await swal({

buttons: {

test: true,

},

});

expect(value).toBe("test");

});

});

describe("loading", () => {

test("shows loading state", async () => {

swal({

button: {

text: "HEPP",

closeModal: false,

},

});

const $button = $(`.${BUTTON}--confirm`);

expect($button.hasClass('swal-button--loading')).toBeFalsy();

$button.click();

expect($button.hasClass('swal-button--loading')).toBeTruthy();

});

});

describe("set class name", () => {

test("sets single class name as string", async () => {

swal({

button: {

text: "TEST",

closeModal: true,

className: 'single-class'

},

});

const $button = $(`.${BUTTON}--confirm`);

expect($button.hasClass('single-class')).toBeTruthy();

});

test("sets multiple class names as string", async () => {

swal({

button: {

text: "TEST",

closeModal: true,

className: 'class1 class2'

},

});

const $button = $(`.${BUTTON}--confirm`);

expect($button.hasClass('class1')).toBeTruthy();

expect($button.hasClass('class2')).toBeTruthy();

});

test("sets multiple class names as array", async () => {

swal({

button: {

text: "TEST",

closeModal: true,

className: ["class1", "class2"]

},

});

const $button = $(`.${BUTTON}--confirm`);

expect($button.hasClass('class1')).toBeTruthy();

expect($button.hasClass('class2')).toBeTruthy();

});

});