import { CONFIRM\_KEY } from './options/buttons';

export interface SwalState {

isOpen: boolean,

promise: {

resolve?(value: string): void,

reject?(): void,

},

actions: {

[namespace: string]: {

value?: string | any,

closeModal?: boolean

},

},

timer: number,

};

export interface ActionOptions {

[buttonNamespace: string]: {

value?: string,

closeModal?: boolean

},

};

const defaultState: SwalState = {

isOpen: false,

promise: null,

actions: {},

timer: null,

};

let state: SwalState = Object.assign({}, defaultState);

export const resetState = (): void => {

state = Object.assign({}, defaultState);

}

/\*

\* Change what the promise resolves to when the user clicks the button.

\* This is called internally when using { input: true } for example.

\*/

export const setActionValue = (opts: string|ActionOptions) => {

if (typeof opts === "string") {

return setActionValueForButton(CONFIRM\_KEY, opts);

}

for (let namespace in opts) {

setActionValueForButton(namespace, opts[namespace]);

}

};

const setActionValueForButton = (namespace: string, value: string | any) => {

if (!state.actions[namespace]) {

state.actions[namespace] = {};

}

Object.assign(state.actions[namespace], {

value,

});

};

/\*

\* Sets other button options, e.g.

\* whether the button should close the modal or not

\*/

export const setActionOptionsFor = (buttonKey: string, {

closeModal = true,

} = {}) => {

Object.assign(state.actions[buttonKey], {

closeModal,

});

};

export default state;