import { ContentOptions } from '../options/content';

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

import { injectElIntoModal } from './modal';

import { contentMarkup } from '../markup';

import { setActionValue } from '../state';

import { onAction } from '../actions';

import CLASS\_NAMES from '../class-list';

const { CONTENT } = CLASS\_NAMES;

/\*

\* Add an <input> to the content container.

\* Update the "promised" value of the confirm button whenever

\* the user types into the input (+ make it "" by default)

\* Set the default focus on the input.

\*/

const addInputEvents = (input: HTMLElement): void => {

input.addEventListener('input', (e) => {

const target = e.target as HTMLInputElement;

const text = target.value;

setActionValue(text);

});

input.addEventListener('keyup', (e) => {

if (e.key === "Enter") {

return onAction(CONFIRM\_KEY);

}

});

/\*

\* FIXME (this is a bit hacky)

\* We're overwriting the default value of confirm button,

\* as well as overwriting the default focus on the button

\*/

setTimeout(() => {

input.focus();

setActionValue('');

}, 0);

};

const initPredefinedContent = (content: Node, elName: string, attrs: any): void => {

const el: HTMLElement = document.createElement(elName);

const elClass = `${CONTENT}\_\_${elName}`;

el.classList.add(elClass);

// Set things like "placeholder":

for (let key in attrs) {

let value: string = attrs[key];

(<any>el)[key] = value;

}

if (elName === "input") {

addInputEvents(el);

}

content.appendChild(el);

};

const initContent = (opts: ContentOptions): void => {

if (!opts) return;

const content: Node = injectElIntoModal(contentMarkup);

const { element, attributes } = opts;

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

initPredefinedContent(content, element, attributes);

} else {

content.appendChild(element);

}

};

export default initContent;