/\*!

\* Bootstrap v4.0.0 (https://getbootstrap.com)

\* Copyright 2011-2018 The Bootstrap Authors (https://github.com/twbs/bootstrap/graphs/contributors)

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\*/

(function (global, factory) {

typeof exports === 'object' && typeof module !== 'undefined' ? factory(exports, require('jquery')) :

typeof define === 'function' && define.amd ? define(['exports', 'jquery'], factory) :

(factory((global.bootstrap = {}),global.jQuery));

}(this, (function (exports,$) { 'use strict';

$ = $ && $.hasOwnProperty('default') ? $['default'] : $;

function \_defineProperties(target, props) {

for (var i = 0; i < props.length; i++) {

var descriptor = props[i];

descriptor.enumerable = descriptor.enumerable || false;

descriptor.configurable = true;

if ("value" in descriptor) descriptor.writable = true;

Object.defineProperty(target, descriptor.key, descriptor);

}

}

function \_createClass(Constructor, protoProps, staticProps) {

if (protoProps) \_defineProperties(Constructor.prototype, protoProps);

if (staticProps) \_defineProperties(Constructor, staticProps);

return Constructor;

}

function \_extends() {

\_extends = Object.assign || function (target) {

for (var i = 1; i < arguments.length; i++) {

var source = arguments[i];

for (var key in source) {

if (Object.prototype.hasOwnProperty.call(source, key)) {

target[key] = source[key];

}

}

}

return target;

};

return \_extends.apply(this, arguments);

}

function \_inheritsLoose(subClass, superClass) {

subClass.prototype = Object.create(superClass.prototype);

subClass.prototype.constructor = subClass;

subClass.\_\_proto\_\_ = superClass;

}

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): util.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Util = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Private TransitionEnd Helpers

\* ------------------------------------------------------------------------

\*/

var transition = false;

var MAX\_UID = 1000000; // Shoutout AngusCroll (https://goo.gl/pxwQGp)

function toType(obj) {

return {}.toString.call(obj).match(/\s([a-zA-Z]+)/)[1].toLowerCase();

}

function getSpecialTransitionEndEvent() {

return {

bindType: transition.end,

delegateType: transition.end,

handle: function handle(event) {

if ($$$1(event.target).is(this)) {

return event.handleObj.handler.apply(this, arguments); // eslint-disable-line prefer-rest-params

}

return undefined; // eslint-disable-line no-undefined

}

};

}

function transitionEndTest() {

if (typeof window !== 'undefined' && window.QUnit) {

return false;

}

return {

end: 'transitionend'

};

}

function transitionEndEmulator(duration) {

var \_this = this;

var called = false;

$$$1(this).one(Util.TRANSITION\_END, function () {

called = true;

});

setTimeout(function () {

if (!called) {

Util.triggerTransitionEnd(\_this);

}

}, duration);

return this;

}

function setTransitionEndSupport() {

transition = transitionEndTest();

$$$1.fn.emulateTransitionEnd = transitionEndEmulator;

if (Util.supportsTransitionEnd()) {

$$$1.event.special[Util.TRANSITION\_END] = getSpecialTransitionEndEvent();

}

}

function escapeId(selector) {

// We escape IDs in case of special selectors (selector = '#myId:something')

// $.escapeSelector does not exist in jQuery < 3

selector = typeof $$$1.escapeSelector === 'function' ? $$$1.escapeSelector(selector).substr(1) : selector.replace(/(:|\.|\[|\]|,|=|@)/g, '\\$1');

return selector;

}

/\*\*

\* --------------------------------------------------------------------------

\* Public Util Api

\* --------------------------------------------------------------------------

\*/

var Util = {

TRANSITION\_END: 'bsTransitionEnd',

getUID: function getUID(prefix) {

do {

// eslint-disable-next-line no-bitwise

prefix += ~~(Math.random() \* MAX\_UID); // "~~" acts like a faster Math.floor() here

} while (document.getElementById(prefix));

return prefix;

},

getSelectorFromElement: function getSelectorFromElement(element) {

var selector = element.getAttribute('data-target');

if (!selector || selector === '#') {

selector = element.getAttribute('href') || '';

} // If it's an ID

if (selector.charAt(0) === '#') {

selector = escapeId(selector);

}

try {

var $selector = $$$1(document).find(selector);

return $selector.length > 0 ? selector : null;

} catch (err) {

return null;

}

},

reflow: function reflow(element) {

return element.offsetHeight;

},

triggerTransitionEnd: function triggerTransitionEnd(element) {

$$$1(element).trigger(transition.end);

},

supportsTransitionEnd: function supportsTransitionEnd() {

return Boolean(transition);

},

isElement: function isElement(obj) {

return (obj[0] || obj).nodeType;

},

typeCheckConfig: function typeCheckConfig(componentName, config, configTypes) {

for (var property in configTypes) {

if (Object.prototype.hasOwnProperty.call(configTypes, property)) {

var expectedTypes = configTypes[property];

var value = config[property];

var valueType = value && Util.isElement(value) ? 'element' : toType(value);

if (!new RegExp(expectedTypes).test(valueType)) {

throw new Error(componentName.toUpperCase() + ": " + ("Option \"" + property + "\" provided type \"" + valueType + "\" ") + ("but expected type \"" + expectedTypes + "\"."));

}

}

}

}

};

setTransitionEndSupport();

return Util;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): alert.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Alert = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'alert';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.alert';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var TRANSITION\_DURATION = 150;

var Selector = {

DISMISS: '[data-dismiss="alert"]'

};

var Event = {

CLOSE: "close" + EVENT\_KEY,

CLOSED: "closed" + EVENT\_KEY,

CLICK\_DATA\_API: "click" + EVENT\_KEY + DATA\_API\_KEY

};

var ClassName = {

ALERT: 'alert',

FADE: 'fade',

SHOW: 'show'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Alert =

/\*#\_\_PURE\_\_\*/

function () {

function Alert(element) {

this.\_element = element;

} // Getters

var \_proto = Alert.prototype;

// Public

\_proto.close = function close(element) {

element = element || this.\_element;

var rootElement = this.\_getRootElement(element);

var customEvent = this.\_triggerCloseEvent(rootElement);

if (customEvent.isDefaultPrevented()) {

return;

}

this.\_removeElement(rootElement);

};

\_proto.dispose = function dispose() {

$$$1.removeData(this.\_element, DATA\_KEY);

this.\_element = null;

}; // Private

\_proto.\_getRootElement = function \_getRootElement(element) {

var selector = Util.getSelectorFromElement(element);

var parent = false;

if (selector) {

parent = $$$1(selector)[0];

}

if (!parent) {

parent = $$$1(element).closest("." + ClassName.ALERT)[0];

}

return parent;

};

\_proto.\_triggerCloseEvent = function \_triggerCloseEvent(element) {

var closeEvent = $$$1.Event(Event.CLOSE);

$$$1(element).trigger(closeEvent);

return closeEvent;

};

\_proto.\_removeElement = function \_removeElement(element) {

var \_this = this;

$$$1(element).removeClass(ClassName.SHOW);

if (!Util.supportsTransitionEnd() || !$$$1(element).hasClass(ClassName.FADE)) {

this.\_destroyElement(element);

return;

}

$$$1(element).one(Util.TRANSITION\_END, function (event) {

return \_this.\_destroyElement(element, event);

}).emulateTransitionEnd(TRANSITION\_DURATION);

};

\_proto.\_destroyElement = function \_destroyElement(element) {

$$$1(element).detach().trigger(Event.CLOSED).remove();

}; // Static

Alert.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var $element = $$$1(this);

var data = $element.data(DATA\_KEY);

if (!data) {

data = new Alert(this);

$element.data(DATA\_KEY, data);

}

if (config === 'close') {

data[config](this);

}

});

};

Alert.\_handleDismiss = function \_handleDismiss(alertInstance) {

return function (event) {

if (event) {

event.preventDefault();

}

alertInstance.close(this);

};

};

\_createClass(Alert, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}]);

return Alert;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(document).on(Event.CLICK\_DATA\_API, Selector.DISMISS, Alert.\_handleDismiss(new Alert()));

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Alert.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Alert;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Alert.\_jQueryInterface;

};

return Alert;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): button.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Button = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'button';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.button';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var ClassName = {

ACTIVE: 'active',

BUTTON: 'btn',

FOCUS: 'focus'

};

var Selector = {

DATA\_TOGGLE\_CARROT: '[data-toggle^="button"]',

DATA\_TOGGLE: '[data-toggle="buttons"]',

INPUT: 'input',

ACTIVE: '.active',

BUTTON: '.btn'

};

var Event = {

CLICK\_DATA\_API: "click" + EVENT\_KEY + DATA\_API\_KEY,

FOCUS\_BLUR\_DATA\_API: "focus" + EVENT\_KEY + DATA\_API\_KEY + " " + ("blur" + EVENT\_KEY + DATA\_API\_KEY)

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Button =

/\*#\_\_PURE\_\_\*/

function () {

function Button(element) {

this.\_element = element;

} // Getters

var \_proto = Button.prototype;

// Public

\_proto.toggle = function toggle() {

var triggerChangeEvent = true;

var addAriaPressed = true;

var rootElement = $$$1(this.\_element).closest(Selector.DATA\_TOGGLE)[0];

if (rootElement) {

var input = $$$1(this.\_element).find(Selector.INPUT)[0];

if (input) {

if (input.type === 'radio') {

if (input.checked && $$$1(this.\_element).hasClass(ClassName.ACTIVE)) {

triggerChangeEvent = false;

} else {

var activeElement = $$$1(rootElement).find(Selector.ACTIVE)[0];

if (activeElement) {

$$$1(activeElement).removeClass(ClassName.ACTIVE);

}

}

}

if (triggerChangeEvent) {

if (input.hasAttribute('disabled') || rootElement.hasAttribute('disabled') || input.classList.contains('disabled') || rootElement.classList.contains('disabled')) {

return;

}

input.checked = !$$$1(this.\_element).hasClass(ClassName.ACTIVE);

$$$1(input).trigger('change');

}

input.focus();

addAriaPressed = false;

}

}

if (addAriaPressed) {

this.\_element.setAttribute('aria-pressed', !$$$1(this.\_element).hasClass(ClassName.ACTIVE));

}

if (triggerChangeEvent) {

$$$1(this.\_element).toggleClass(ClassName.ACTIVE);

}

};

\_proto.dispose = function dispose() {

$$$1.removeData(this.\_element, DATA\_KEY);

this.\_element = null;

}; // Static

Button.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $$$1(this).data(DATA\_KEY);

if (!data) {

data = new Button(this);

$$$1(this).data(DATA\_KEY, data);

}

if (config === 'toggle') {

data[config]();

}

});

};

\_createClass(Button, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}]);

return Button;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(document).on(Event.CLICK\_DATA\_API, Selector.DATA\_TOGGLE\_CARROT, function (event) {

event.preventDefault();

var button = event.target;

if (!$$$1(button).hasClass(ClassName.BUTTON)) {

button = $$$1(button).closest(Selector.BUTTON);

}

Button.\_jQueryInterface.call($$$1(button), 'toggle');

}).on(Event.FOCUS\_BLUR\_DATA\_API, Selector.DATA\_TOGGLE\_CARROT, function (event) {

var button = $$$1(event.target).closest(Selector.BUTTON)[0];

$$$1(button).toggleClass(ClassName.FOCUS, /^focus(in)?$/.test(event.type));

});

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Button.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Button;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Button.\_jQueryInterface;

};

return Button;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): carousel.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Carousel = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'carousel';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.carousel';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var TRANSITION\_DURATION = 600;

var ARROW\_LEFT\_KEYCODE = 37; // KeyboardEvent.which value for left arrow key

var ARROW\_RIGHT\_KEYCODE = 39; // KeyboardEvent.which value for right arrow key

var TOUCHEVENT\_COMPAT\_WAIT = 500; // Time for mouse compat events to fire after touch

var Default = {

interval: 5000,

keyboard: true,

slide: false,

pause: 'hover',

wrap: true

};

var DefaultType = {

interval: '(number|boolean)',

keyboard: 'boolean',

slide: '(boolean|string)',

pause: '(string|boolean)',

wrap: 'boolean'

};

var Direction = {

NEXT: 'next',

PREV: 'prev',

LEFT: 'left',

RIGHT: 'right'

};

var Event = {

SLIDE: "slide" + EVENT\_KEY,

SLID: "slid" + EVENT\_KEY,

KEYDOWN: "keydown" + EVENT\_KEY,

MOUSEENTER: "mouseenter" + EVENT\_KEY,

MOUSELEAVE: "mouseleave" + EVENT\_KEY,

TOUCHEND: "touchend" + EVENT\_KEY,

LOAD\_DATA\_API: "load" + EVENT\_KEY + DATA\_API\_KEY,

CLICK\_DATA\_API: "click" + EVENT\_KEY + DATA\_API\_KEY

};

var ClassName = {

CAROUSEL: 'carousel',

ACTIVE: 'active',

SLIDE: 'slide',

RIGHT: 'carousel-item-right',

LEFT: 'carousel-item-left',

NEXT: 'carousel-item-next',

PREV: 'carousel-item-prev',

ITEM: 'carousel-item'

};

var Selector = {

ACTIVE: '.active',

ACTIVE\_ITEM: '.active.carousel-item',

ITEM: '.carousel-item',

NEXT\_PREV: '.carousel-item-next, .carousel-item-prev',

INDICATORS: '.carousel-indicators',

DATA\_SLIDE: '[data-slide], [data-slide-to]',

DATA\_RIDE: '[data-ride="carousel"]'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Carousel =

/\*#\_\_PURE\_\_\*/

function () {

function Carousel(element, config) {

this.\_items = null;

this.\_interval = null;

this.\_activeElement = null;

this.\_isPaused = false;

this.\_isSliding = false;

this.touchTimeout = null;

this.\_config = this.\_getConfig(config);

this.\_element = $$$1(element)[0];

this.\_indicatorsElement = $$$1(this.\_element).find(Selector.INDICATORS)[0];

this.\_addEventListeners();

} // Getters

var \_proto = Carousel.prototype;

// Public

\_proto.next = function next() {

if (!this.\_isSliding) {

this.\_slide(Direction.NEXT);

}

};

\_proto.nextWhenVisible = function nextWhenVisible() {

// Don't call next when the page isn't visible

// or the carousel or its parent isn't visible

if (!document.hidden && $$$1(this.\_element).is(':visible') && $$$1(this.\_element).css('visibility') !== 'hidden') {

this.next();

}

};

\_proto.prev = function prev() {

if (!this.\_isSliding) {

this.\_slide(Direction.PREV);

}

};

\_proto.pause = function pause(event) {

if (!event) {

this.\_isPaused = true;

}

if ($$$1(this.\_element).find(Selector.NEXT\_PREV)[0] && Util.supportsTransitionEnd()) {

Util.triggerTransitionEnd(this.\_element);

this.cycle(true);

}

clearInterval(this.\_interval);

this.\_interval = null;

};

\_proto.cycle = function cycle(event) {

if (!event) {

this.\_isPaused = false;

}

if (this.\_interval) {

clearInterval(this.\_interval);

this.\_interval = null;

}

if (this.\_config.interval && !this.\_isPaused) {

this.\_interval = setInterval((document.visibilityState ? this.nextWhenVisible : this.next).bind(this), this.\_config.interval);

}

};

\_proto.to = function to(index) {

var \_this = this;

this.\_activeElement = $$$1(this.\_element).find(Selector.ACTIVE\_ITEM)[0];

var activeIndex = this.\_getItemIndex(this.\_activeElement);

if (index > this.\_items.length - 1 || index < 0) {

return;

}

if (this.\_isSliding) {

$$$1(this.\_element).one(Event.SLID, function () {

return \_this.to(index);

});

return;

}

if (activeIndex === index) {

this.pause();

this.cycle();

return;

}

var direction = index > activeIndex ? Direction.NEXT : Direction.PREV;

this.\_slide(direction, this.\_items[index]);

};

\_proto.dispose = function dispose() {

$$$1(this.\_element).off(EVENT\_KEY);

$$$1.removeData(this.\_element, DATA\_KEY);

this.\_items = null;

this.\_config = null;

this.\_element = null;

this.\_interval = null;

this.\_isPaused = null;

this.\_isSliding = null;

this.\_activeElement = null;

this.\_indicatorsElement = null;

}; // Private

\_proto.\_getConfig = function \_getConfig(config) {

config = \_extends({}, Default, config);

Util.typeCheckConfig(NAME, config, DefaultType);

return config;

};

\_proto.\_addEventListeners = function \_addEventListeners() {

var \_this2 = this;

if (this.\_config.keyboard) {

$$$1(this.\_element).on(Event.KEYDOWN, function (event) {

return \_this2.\_keydown(event);

});

}

if (this.\_config.pause === 'hover') {

$$$1(this.\_element).on(Event.MOUSEENTER, function (event) {

return \_this2.pause(event);

}).on(Event.MOUSELEAVE, function (event) {

return \_this2.cycle(event);

});

if ('ontouchstart' in document.documentElement) {

// If it's a touch-enabled device, mouseenter/leave are fired as

// part of the mouse compatibility events on first tap - the carousel

// would stop cycling until user tapped out of it;

// here, we listen for touchend, explicitly pause the carousel

// (as if it's the second time we tap on it, mouseenter compat event

// is NOT fired) and after a timeout (to allow for mouse compatibility

// events to fire) we explicitly restart cycling

$$$1(this.\_element).on(Event.TOUCHEND, function () {

\_this2.pause();

if (\_this2.touchTimeout) {

clearTimeout(\_this2.touchTimeout);

}

\_this2.touchTimeout = setTimeout(function (event) {

return \_this2.cycle(event);

}, TOUCHEVENT\_COMPAT\_WAIT + \_this2.\_config.interval);

});

}

}

};

\_proto.\_keydown = function \_keydown(event) {

if (/input|textarea/i.test(event.target.tagName)) {

return;

}

switch (event.which) {

case ARROW\_LEFT\_KEYCODE:

event.preventDefault();

this.prev();

break;

case ARROW\_RIGHT\_KEYCODE:

event.preventDefault();

this.next();

break;

default:

}

};

\_proto.\_getItemIndex = function \_getItemIndex(element) {

this.\_items = $$$1.makeArray($$$1(element).parent().find(Selector.ITEM));

return this.\_items.indexOf(element);

};

\_proto.\_getItemByDirection = function \_getItemByDirection(direction, activeElement) {

var isNextDirection = direction === Direction.NEXT;

var isPrevDirection = direction === Direction.PREV;

var activeIndex = this.\_getItemIndex(activeElement);

var lastItemIndex = this.\_items.length - 1;

var isGoingToWrap = isPrevDirection && activeIndex === 0 || isNextDirection && activeIndex === lastItemIndex;

if (isGoingToWrap && !this.\_config.wrap) {

return activeElement;

}

var delta = direction === Direction.PREV ? -1 : 1;

var itemIndex = (activeIndex + delta) % this.\_items.length;

return itemIndex === -1 ? this.\_items[this.\_items.length - 1] : this.\_items[itemIndex];

};

\_proto.\_triggerSlideEvent = function \_triggerSlideEvent(relatedTarget, eventDirectionName) {

var targetIndex = this.\_getItemIndex(relatedTarget);

var fromIndex = this.\_getItemIndex($$$1(this.\_element).find(Selector.ACTIVE\_ITEM)[0]);

var slideEvent = $$$1.Event(Event.SLIDE, {

relatedTarget: relatedTarget,

direction: eventDirectionName,

from: fromIndex,

to: targetIndex

});

$$$1(this.\_element).trigger(slideEvent);

return slideEvent;

};

\_proto.\_setActiveIndicatorElement = function \_setActiveIndicatorElement(element) {

if (this.\_indicatorsElement) {

$$$1(this.\_indicatorsElement).find(Selector.ACTIVE).removeClass(ClassName.ACTIVE);

var nextIndicator = this.\_indicatorsElement.children[this.\_getItemIndex(element)];

if (nextIndicator) {

$$$1(nextIndicator).addClass(ClassName.ACTIVE);

}

}

};

\_proto.\_slide = function \_slide(direction, element) {

var \_this3 = this;

var activeElement = $$$1(this.\_element).find(Selector.ACTIVE\_ITEM)[0];

var activeElementIndex = this.\_getItemIndex(activeElement);

var nextElement = element || activeElement && this.\_getItemByDirection(direction, activeElement);

var nextElementIndex = this.\_getItemIndex(nextElement);

var isCycling = Boolean(this.\_interval);

var directionalClassName;

var orderClassName;

var eventDirectionName;

if (direction === Direction.NEXT) {

directionalClassName = ClassName.LEFT;

orderClassName = ClassName.NEXT;

eventDirectionName = Direction.LEFT;

} else {

directionalClassName = ClassName.RIGHT;

orderClassName = ClassName.PREV;

eventDirectionName = Direction.RIGHT;

}

if (nextElement && $$$1(nextElement).hasClass(ClassName.ACTIVE)) {

this.\_isSliding = false;

return;

}

var slideEvent = this.\_triggerSlideEvent(nextElement, eventDirectionName);

if (slideEvent.isDefaultPrevented()) {

return;

}

if (!activeElement || !nextElement) {

// Some weirdness is happening, so we bail

return;

}

this.\_isSliding = true;

if (isCycling) {

this.pause();

}

this.\_setActiveIndicatorElement(nextElement);

var slidEvent = $$$1.Event(Event.SLID, {

relatedTarget: nextElement,

direction: eventDirectionName,

from: activeElementIndex,

to: nextElementIndex

});

if (Util.supportsTransitionEnd() && $$$1(this.\_element).hasClass(ClassName.SLIDE)) {

$$$1(nextElement).addClass(orderClassName);

Util.reflow(nextElement);

$$$1(activeElement).addClass(directionalClassName);

$$$1(nextElement).addClass(directionalClassName);

$$$1(activeElement).one(Util.TRANSITION\_END, function () {

$$$1(nextElement).removeClass(directionalClassName + " " + orderClassName).addClass(ClassName.ACTIVE);

$$$1(activeElement).removeClass(ClassName.ACTIVE + " " + orderClassName + " " + directionalClassName);

\_this3.\_isSliding = false;

setTimeout(function () {

return $$$1(\_this3.\_element).trigger(slidEvent);

}, 0);

}).emulateTransitionEnd(TRANSITION\_DURATION);

} else {

$$$1(activeElement).removeClass(ClassName.ACTIVE);

$$$1(nextElement).addClass(ClassName.ACTIVE);

this.\_isSliding = false;

$$$1(this.\_element).trigger(slidEvent);

}

if (isCycling) {

this.cycle();

}

}; // Static

Carousel.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $$$1(this).data(DATA\_KEY);

var \_config = \_extends({}, Default, $$$1(this).data());

if (typeof config === 'object') {

\_config = \_extends({}, \_config, config);

}

var action = typeof config === 'string' ? config : \_config.slide;

if (!data) {

data = new Carousel(this, \_config);

$$$1(this).data(DATA\_KEY, data);

}

if (typeof config === 'number') {

data.to(config);

} else if (typeof action === 'string') {

if (typeof data[action] === 'undefined') {

throw new TypeError("No method named \"" + action + "\"");

}

data[action]();

} else if (\_config.interval) {

data.pause();

data.cycle();

}

});

};

Carousel.\_dataApiClickHandler = function \_dataApiClickHandler(event) {

var selector = Util.getSelectorFromElement(this);

if (!selector) {

return;

}

var target = $$$1(selector)[0];

if (!target || !$$$1(target).hasClass(ClassName.CAROUSEL)) {

return;

}

var config = \_extends({}, $$$1(target).data(), $$$1(this).data());

var slideIndex = this.getAttribute('data-slide-to');

if (slideIndex) {

config.interval = false;

}

Carousel.\_jQueryInterface.call($$$1(target), config);

if (slideIndex) {

$$$1(target).data(DATA\_KEY).to(slideIndex);

}

event.preventDefault();

};

\_createClass(Carousel, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}, {

key: "Default",

get: function get() {

return Default;

}

}]);

return Carousel;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(document).on(Event.CLICK\_DATA\_API, Selector.DATA\_SLIDE, Carousel.\_dataApiClickHandler);

$$$1(window).on(Event.LOAD\_DATA\_API, function () {

$$$1(Selector.DATA\_RIDE).each(function () {

var $carousel = $$$1(this);

Carousel.\_jQueryInterface.call($carousel, $carousel.data());

});

});

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Carousel.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Carousel;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Carousel.\_jQueryInterface;

};

return Carousel;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): collapse.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Collapse = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'collapse';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.collapse';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var TRANSITION\_DURATION = 600;

var Default = {

toggle: true,

parent: ''

};

var DefaultType = {

toggle: 'boolean',

parent: '(string|element)'

};

var Event = {

SHOW: "show" + EVENT\_KEY,

SHOWN: "shown" + EVENT\_KEY,

HIDE: "hide" + EVENT\_KEY,

HIDDEN: "hidden" + EVENT\_KEY,

CLICK\_DATA\_API: "click" + EVENT\_KEY + DATA\_API\_KEY

};

var ClassName = {

SHOW: 'show',

COLLAPSE: 'collapse',

COLLAPSING: 'collapsing',

COLLAPSED: 'collapsed'

};

var Dimension = {

WIDTH: 'width',

HEIGHT: 'height'

};

var Selector = {

ACTIVES: '.show, .collapsing',

DATA\_TOGGLE: '[data-toggle="collapse"]'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Collapse =

/\*#\_\_PURE\_\_\*/

function () {

function Collapse(element, config) {

this.\_isTransitioning = false;

this.\_element = element;

this.\_config = this.\_getConfig(config);

this.\_triggerArray = $$$1.makeArray($$$1("[data-toggle=\"collapse\"][href=\"#" + element.id + "\"]," + ("[data-toggle=\"collapse\"][data-target=\"#" + element.id + "\"]")));

var tabToggles = $$$1(Selector.DATA\_TOGGLE);

for (var i = 0; i < tabToggles.length; i++) {

var elem = tabToggles[i];

var selector = Util.getSelectorFromElement(elem);

if (selector !== null && $$$1(selector).filter(element).length > 0) {

this.\_selector = selector;

this.\_triggerArray.push(elem);

}

}

this.\_parent = this.\_config.parent ? this.\_getParent() : null;

if (!this.\_config.parent) {

this.\_addAriaAndCollapsedClass(this.\_element, this.\_triggerArray);

}

if (this.\_config.toggle) {

this.toggle();

}

} // Getters

var \_proto = Collapse.prototype;

// Public

\_proto.toggle = function toggle() {

if ($$$1(this.\_element).hasClass(ClassName.SHOW)) {

this.hide();

} else {

this.show();

}

};

\_proto.show = function show() {

var \_this = this;

if (this.\_isTransitioning || $$$1(this.\_element).hasClass(ClassName.SHOW)) {

return;

}

var actives;

var activesData;

if (this.\_parent) {

actives = $$$1.makeArray($$$1(this.\_parent).find(Selector.ACTIVES).filter("[data-parent=\"" + this.\_config.parent + "\"]"));

if (actives.length === 0) {

actives = null;

}

}

if (actives) {

activesData = $$$1(actives).not(this.\_selector).data(DATA\_KEY);

if (activesData && activesData.\_isTransitioning) {

return;

}

}

var startEvent = $$$1.Event(Event.SHOW);

$$$1(this.\_element).trigger(startEvent);

if (startEvent.isDefaultPrevented()) {

return;

}

if (actives) {

Collapse.\_jQueryInterface.call($$$1(actives).not(this.\_selector), 'hide');

if (!activesData) {

$$$1(actives).data(DATA\_KEY, null);

}

}

var dimension = this.\_getDimension();

$$$1(this.\_element).removeClass(ClassName.COLLAPSE).addClass(ClassName.COLLAPSING);

this.\_element.style[dimension] = 0;

if (this.\_triggerArray.length > 0) {

$$$1(this.\_triggerArray).removeClass(ClassName.COLLAPSED).attr('aria-expanded', true);

}

this.setTransitioning(true);

var complete = function complete() {

$$$1(\_this.\_element).removeClass(ClassName.COLLAPSING).addClass(ClassName.COLLAPSE).addClass(ClassName.SHOW);

\_this.\_element.style[dimension] = '';

\_this.setTransitioning(false);

$$$1(\_this.\_element).trigger(Event.SHOWN);

};

if (!Util.supportsTransitionEnd()) {

complete();

return;

}

var capitalizedDimension = dimension[0].toUpperCase() + dimension.slice(1);

var scrollSize = "scroll" + capitalizedDimension;

$$$1(this.\_element).one(Util.TRANSITION\_END, complete).emulateTransitionEnd(TRANSITION\_DURATION);

this.\_element.style[dimension] = this.\_element[scrollSize] + "px";

};

\_proto.hide = function hide() {

var \_this2 = this;

if (this.\_isTransitioning || !$$$1(this.\_element).hasClass(ClassName.SHOW)) {

return;

}

var startEvent = $$$1.Event(Event.HIDE);

$$$1(this.\_element).trigger(startEvent);

if (startEvent.isDefaultPrevented()) {

return;

}

var dimension = this.\_getDimension();

this.\_element.style[dimension] = this.\_element.getBoundingClientRect()[dimension] + "px";

Util.reflow(this.\_element);

$$$1(this.\_element).addClass(ClassName.COLLAPSING).removeClass(ClassName.COLLAPSE).removeClass(ClassName.SHOW);

if (this.\_triggerArray.length > 0) {

for (var i = 0; i < this.\_triggerArray.length; i++) {

var trigger = this.\_triggerArray[i];

var selector = Util.getSelectorFromElement(trigger);

if (selector !== null) {

var $elem = $$$1(selector);

if (!$elem.hasClass(ClassName.SHOW)) {

$$$1(trigger).addClass(ClassName.COLLAPSED).attr('aria-expanded', false);

}

}

}

}

this.setTransitioning(true);

var complete = function complete() {

\_this2.setTransitioning(false);

$$$1(\_this2.\_element).removeClass(ClassName.COLLAPSING).addClass(ClassName.COLLAPSE).trigger(Event.HIDDEN);

};

this.\_element.style[dimension] = '';

if (!Util.supportsTransitionEnd()) {

complete();

return;

}

$$$1(this.\_element).one(Util.TRANSITION\_END, complete).emulateTransitionEnd(TRANSITION\_DURATION);

};

\_proto.setTransitioning = function setTransitioning(isTransitioning) {

this.\_isTransitioning = isTransitioning;

};

\_proto.dispose = function dispose() {

$$$1.removeData(this.\_element, DATA\_KEY);

this.\_config = null;

this.\_parent = null;

this.\_element = null;

this.\_triggerArray = null;

this.\_isTransitioning = null;

}; // Private

\_proto.\_getConfig = function \_getConfig(config) {

config = \_extends({}, Default, config);

config.toggle = Boolean(config.toggle); // Coerce string values

Util.typeCheckConfig(NAME, config, DefaultType);

return config;

};

\_proto.\_getDimension = function \_getDimension() {

var hasWidth = $$$1(this.\_element).hasClass(Dimension.WIDTH);

return hasWidth ? Dimension.WIDTH : Dimension.HEIGHT;

};

\_proto.\_getParent = function \_getParent() {

var \_this3 = this;

var parent = null;

if (Util.isElement(this.\_config.parent)) {

parent = this.\_config.parent; // It's a jQuery object

if (typeof this.\_config.parent.jquery !== 'undefined') {

parent = this.\_config.parent[0];

}

} else {

parent = $$$1(this.\_config.parent)[0];

}

var selector = "[data-toggle=\"collapse\"][data-parent=\"" + this.\_config.parent + "\"]";

$$$1(parent).find(selector).each(function (i, element) {

\_this3.\_addAriaAndCollapsedClass(Collapse.\_getTargetFromElement(element), [element]);

});

return parent;

};

\_proto.\_addAriaAndCollapsedClass = function \_addAriaAndCollapsedClass(element, triggerArray) {

if (element) {

var isOpen = $$$1(element).hasClass(ClassName.SHOW);

if (triggerArray.length > 0) {

$$$1(triggerArray).toggleClass(ClassName.COLLAPSED, !isOpen).attr('aria-expanded', isOpen);

}

}

}; // Static

Collapse.\_getTargetFromElement = function \_getTargetFromElement(element) {

var selector = Util.getSelectorFromElement(element);

return selector ? $$$1(selector)[0] : null;

};

Collapse.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var $this = $$$1(this);

var data = $this.data(DATA\_KEY);

var \_config = \_extends({}, Default, $this.data(), typeof config === 'object' && config);

if (!data && \_config.toggle && /show|hide/.test(config)) {

\_config.toggle = false;

}

if (!data) {

data = new Collapse(this, \_config);

$this.data(DATA\_KEY, data);

}

if (typeof config === 'string') {

if (typeof data[config] === 'undefined') {

throw new TypeError("No method named \"" + config + "\"");

}

data[config]();

}

});

};

\_createClass(Collapse, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}, {

key: "Default",

get: function get() {

return Default;

}

}]);

return Collapse;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(document).on(Event.CLICK\_DATA\_API, Selector.DATA\_TOGGLE, function (event) {

// preventDefault only for <a> elements (which change the URL) not inside the collapsible element

if (event.currentTarget.tagName === 'A') {

event.preventDefault();

}

var $trigger = $$$1(this);

var selector = Util.getSelectorFromElement(this);

$$$1(selector).each(function () {

var $target = $$$1(this);

var data = $target.data(DATA\_KEY);

var config = data ? 'toggle' : $trigger.data();

Collapse.\_jQueryInterface.call($target, config);

});

});

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Collapse.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Collapse;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Collapse.\_jQueryInterface;

};

return Collapse;

}($);

/\*\*!

\* @fileOverview Kickass library to create and place poppers near their reference elements.

\* @version 1.12.9

\* @license

\* Copyright (c) 2016 Federico Zivolo and contributors

\*

\* Permission is hereby granted, free of charge, to any person obtaining a copy

\* of this software and associated documentation files (the "Software"), to deal

\* in the Software without restriction, including without limitation the rights

\* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell

\* copies of the Software, and to permit persons to whom the Software is

\* furnished to do so, subject to the following conditions:

\*

\* The above copyright notice and this permission notice shall be included in all

\* copies or substantial portions of the Software.

\*

\* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR

\* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,

\* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE

\* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER

\* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,

\* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE

\* SOFTWARE.

\*/

var isBrowser = typeof window !== 'undefined' && typeof document !== 'undefined';

var longerTimeoutBrowsers = ['Edge', 'Trident', 'Firefox'];

var timeoutDuration = 0;

for (var i = 0; i < longerTimeoutBrowsers.length; i += 1) {

if (isBrowser && navigator.userAgent.indexOf(longerTimeoutBrowsers[i]) >= 0) {

timeoutDuration = 1;

break;

}

}

function microtaskDebounce(fn) {

var called = false;

return function () {

if (called) {

return;

}

called = true;

window.Promise.resolve().then(function () {

called = false;

fn();

});

};

}

function taskDebounce(fn) {

var scheduled = false;

return function () {

if (!scheduled) {

scheduled = true;

setTimeout(function () {

scheduled = false;

fn();

}, timeoutDuration);

}

};

}

var supportsMicroTasks = isBrowser && window.Promise;

/\*\*

\* Create a debounced version of a method, that's asynchronously deferred

\* but called in the minimum time possible.

\*

\* @method

\* @memberof Popper.Utils

\* @argument {Function} fn

\* @returns {Function}

\*/

var debounce = supportsMicroTasks ? microtaskDebounce : taskDebounce;

/\*\*

\* Check if the given variable is a function

\* @method

\* @memberof Popper.Utils

\* @argument {Any} functionToCheck - variable to check

\* @returns {Boolean} answer to: is a function?

\*/

function isFunction(functionToCheck) {

var getType = {};

return functionToCheck && getType.toString.call(functionToCheck) === '[object Function]';

}

/\*\*

\* Get CSS computed property of the given element

\* @method

\* @memberof Popper.Utils

\* @argument {Eement} element

\* @argument {String} property

\*/

function getStyleComputedProperty(element, property) {

if (element.nodeType !== 1) {

return [];

}

// NOTE: 1 DOM access here

var css = getComputedStyle(element, null);

return property ? css[property] : css;

}

/\*\*

\* Returns the parentNode or the host of the element

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element

\* @returns {Element} parent

\*/

function getParentNode(element) {

if (element.nodeName === 'HTML') {

return element;

}

return element.parentNode || element.host;

}

/\*\*

\* Returns the scrolling parent of the given element

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element

\* @returns {Element} scroll parent

\*/

function getScrollParent(element) {

// Return body, `getScroll` will take care to get the correct `scrollTop` from it

if (!element) {

return document.body;

}

switch (element.nodeName) {

case 'HTML':

case 'BODY':

return element.ownerDocument.body;

case '#document':

return element.body;

}

// Firefox want us to check `-x` and `-y` variations as well

var \_getStyleComputedProp = getStyleComputedProperty(element),

overflow = \_getStyleComputedProp.overflow,

overflowX = \_getStyleComputedProp.overflowX,

overflowY = \_getStyleComputedProp.overflowY;

if (/(auto|scroll)/.test(overflow + overflowY + overflowX)) {

return element;

}

return getScrollParent(getParentNode(element));

}

/\*\*

\* Returns the offset parent of the given element

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element

\* @returns {Element} offset parent

\*/

function getOffsetParent(element) {

// NOTE: 1 DOM access here

var offsetParent = element && element.offsetParent;

var nodeName = offsetParent && offsetParent.nodeName;

if (!nodeName || nodeName === 'BODY' || nodeName === 'HTML') {

if (element) {

return element.ownerDocument.documentElement;

}

return document.documentElement;

}

// .offsetParent will return the closest TD or TABLE in case

// no offsetParent is present, I hate this job...

if (['TD', 'TABLE'].indexOf(offsetParent.nodeName) !== -1 && getStyleComputedProperty(offsetParent, 'position') === 'static') {

return getOffsetParent(offsetParent);

}

return offsetParent;

}

function isOffsetContainer(element) {

var nodeName = element.nodeName;

if (nodeName === 'BODY') {

return false;

}

return nodeName === 'HTML' || getOffsetParent(element.firstElementChild) === element;

}

/\*\*

\* Finds the root node (document, shadowDOM root) of the given element

\* @method

\* @memberof Popper.Utils

\* @argument {Element} node

\* @returns {Element} root node

\*/

function getRoot(node) {

if (node.parentNode !== null) {

return getRoot(node.parentNode);

}

return node;

}

/\*\*

\* Finds the offset parent common to the two provided nodes

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element1

\* @argument {Element} element2

\* @returns {Element} common offset parent

\*/

function findCommonOffsetParent(element1, element2) {

// This check is needed to avoid errors in case one of the elements isn't defined for any reason

if (!element1 || !element1.nodeType || !element2 || !element2.nodeType) {

return document.documentElement;

}

// Here we make sure to give as "start" the element that comes first in the DOM

var order = element1.compareDocumentPosition(element2) & Node.DOCUMENT\_POSITION\_FOLLOWING;

var start = order ? element1 : element2;

var end = order ? element2 : element1;

// Get common ancestor container

var range = document.createRange();

range.setStart(start, 0);

range.setEnd(end, 0);

var commonAncestorContainer = range.commonAncestorContainer;

// Both nodes are inside #document

if (element1 !== commonAncestorContainer && element2 !== commonAncestorContainer || start.contains(end)) {

if (isOffsetContainer(commonAncestorContainer)) {

return commonAncestorContainer;

}

return getOffsetParent(commonAncestorContainer);

}

// one of the nodes is inside shadowDOM, find which one

var element1root = getRoot(element1);

if (element1root.host) {

return findCommonOffsetParent(element1root.host, element2);

} else {

return findCommonOffsetParent(element1, getRoot(element2).host);

}

}

/\*\*

\* Gets the scroll value of the given element in the given side (top and left)

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element

\* @argument {String} side `top` or `left`

\* @returns {number} amount of scrolled pixels

\*/

function getScroll(element) {

var side = arguments.length > 1 && arguments[1] !== undefined ? arguments[1] : 'top';

var upperSide = side === 'top' ? 'scrollTop' : 'scrollLeft';

var nodeName = element.nodeName;

if (nodeName === 'BODY' || nodeName === 'HTML') {

var html = element.ownerDocument.documentElement;

var scrollingElement = element.ownerDocument.scrollingElement || html;

return scrollingElement[upperSide];

}

return element[upperSide];

}

/\*

\* Sum or subtract the element scroll values (left and top) from a given rect object

\* @method

\* @memberof Popper.Utils

\* @param {Object} rect - Rect object you want to change

\* @param {HTMLElement} element - The element from the function reads the scroll values

\* @param {Boolean} subtract - set to true if you want to subtract the scroll values

\* @return {Object} rect - The modifier rect object

\*/

function includeScroll(rect, element) {

var subtract = arguments.length > 2 && arguments[2] !== undefined ? arguments[2] : false;

var scrollTop = getScroll(element, 'top');

var scrollLeft = getScroll(element, 'left');

var modifier = subtract ? -1 : 1;

rect.top += scrollTop \* modifier;

rect.bottom += scrollTop \* modifier;

rect.left += scrollLeft \* modifier;

rect.right += scrollLeft \* modifier;

return rect;

}

/\*

\* Helper to detect borders of a given element

\* @method

\* @memberof Popper.Utils

\* @param {CSSStyleDeclaration} styles

\* Result of `getStyleComputedProperty` on the given element

\* @param {String} axis - `x` or `y`

\* @return {number} borders - The borders size of the given axis

\*/

function getBordersSize(styles, axis) {

var sideA = axis === 'x' ? 'Left' : 'Top';

var sideB = sideA === 'Left' ? 'Right' : 'Bottom';

return parseFloat(styles['border' + sideA + 'Width'], 10) + parseFloat(styles['border' + sideB + 'Width'], 10);

}

/\*\*

\* Tells if you are running Internet Explorer 10

\* @method

\* @memberof Popper.Utils

\* @returns {Boolean} isIE10

\*/

var isIE10 = undefined;

var isIE10$1 = function () {

if (isIE10 === undefined) {

isIE10 = navigator.appVersion.indexOf('MSIE 10') !== -1;

}

return isIE10;

};

function getSize(axis, body, html, computedStyle) {

return Math.max(body['offset' + axis], body['scroll' + axis], html['client' + axis], html['offset' + axis], html['scroll' + axis], isIE10$1() ? html['offset' + axis] + computedStyle['margin' + (axis === 'Height' ? 'Top' : 'Left')] + computedStyle['margin' + (axis === 'Height' ? 'Bottom' : 'Right')] : 0);

}

function getWindowSizes() {

var body = document.body;

var html = document.documentElement;

var computedStyle = isIE10$1() && getComputedStyle(html);

return {

height: getSize('Height', body, html, computedStyle),

width: getSize('Width', body, html, computedStyle)

};

}

var classCallCheck = function (instance, Constructor) {

if (!(instance instanceof Constructor)) {

throw new TypeError("Cannot call a class as a function");

}

};

var createClass = function () {

function defineProperties(target, props) {

for (var i = 0; i < props.length; i++) {

var descriptor = props[i];

descriptor.enumerable = descriptor.enumerable || false;

descriptor.configurable = true;

if ("value" in descriptor) descriptor.writable = true;

Object.defineProperty(target, descriptor.key, descriptor);

}

}

return function (Constructor, protoProps, staticProps) {

if (protoProps) defineProperties(Constructor.prototype, protoProps);

if (staticProps) defineProperties(Constructor, staticProps);

return Constructor;

};

}();

var defineProperty = function (obj, key, value) {

if (key in obj) {

Object.defineProperty(obj, key, {

value: value,

enumerable: true,

configurable: true,

writable: true

});

} else {

obj[key] = value;

}

return obj;

};

var \_extends$1 = Object.assign || function (target) {

for (var i = 1; i < arguments.length; i++) {

var source = arguments[i];

for (var key in source) {

if (Object.prototype.hasOwnProperty.call(source, key)) {

target[key] = source[key];

}

}

}

return target;

};

/\*\*

\* Given element offsets, generate an output similar to getBoundingClientRect

\* @method

\* @memberof Popper.Utils

\* @argument {Object} offsets

\* @returns {Object} ClientRect like output

\*/

function getClientRect(offsets) {

return \_extends$1({}, offsets, {

right: offsets.left + offsets.width,

bottom: offsets.top + offsets.height

});

}

/\*\*

\* Get bounding client rect of given element

\* @method

\* @memberof Popper.Utils

\* @param {HTMLElement} element

\* @return {Object} client rect

\*/

function getBoundingClientRect(element) {

var rect = {};

// IE10 10 FIX: Please, don't ask, the element isn't

// considered in DOM in some circumstances...

// This isn't reproducible in IE10 compatibility mode of IE11

if (isIE10$1()) {

try {

rect = element.getBoundingClientRect();

var scrollTop = getScroll(element, 'top');

var scrollLeft = getScroll(element, 'left');

rect.top += scrollTop;

rect.left += scrollLeft;

rect.bottom += scrollTop;

rect.right += scrollLeft;

} catch (err) {}

} else {

rect = element.getBoundingClientRect();

}

var result = {

left: rect.left,

top: rect.top,

width: rect.right - rect.left,

height: rect.bottom - rect.top

};

// subtract scrollbar size from sizes

var sizes = element.nodeName === 'HTML' ? getWindowSizes() : {};

var width = sizes.width || element.clientWidth || result.right - result.left;

var height = sizes.height || element.clientHeight || result.bottom - result.top;

var horizScrollbar = element.offsetWidth - width;

var vertScrollbar = element.offsetHeight - height;

// if an hypothetical scrollbar is detected, we must be sure it's not a `border`

// we make this check conditional for performance reasons

if (horizScrollbar || vertScrollbar) {

var styles = getStyleComputedProperty(element);

horizScrollbar -= getBordersSize(styles, 'x');

vertScrollbar -= getBordersSize(styles, 'y');

result.width -= horizScrollbar;

result.height -= vertScrollbar;

}

return getClientRect(result);

}

function getOffsetRectRelativeToArbitraryNode(children, parent) {

var isIE10 = isIE10$1();

var isHTML = parent.nodeName === 'HTML';

var childrenRect = getBoundingClientRect(children);

var parentRect = getBoundingClientRect(parent);

var scrollParent = getScrollParent(children);

var styles = getStyleComputedProperty(parent);

var borderTopWidth = parseFloat(styles.borderTopWidth, 10);

var borderLeftWidth = parseFloat(styles.borderLeftWidth, 10);

var offsets = getClientRect({

top: childrenRect.top - parentRect.top - borderTopWidth,

left: childrenRect.left - parentRect.left - borderLeftWidth,

width: childrenRect.width,

height: childrenRect.height

});

offsets.marginTop = 0;

offsets.marginLeft = 0;

// Subtract margins of documentElement in case it's being used as parent

// we do this only on HTML because it's the only element that behaves

// differently when margins are applied to it. The margins are included in

// the box of the documentElement, in the other cases not.

if (!isIE10 && isHTML) {

var marginTop = parseFloat(styles.marginTop, 10);

var marginLeft = parseFloat(styles.marginLeft, 10);

offsets.top -= borderTopWidth - marginTop;

offsets.bottom -= borderTopWidth - marginTop;

offsets.left -= borderLeftWidth - marginLeft;

offsets.right -= borderLeftWidth - marginLeft;

// Attach marginTop and marginLeft because in some circumstances we may need them

offsets.marginTop = marginTop;

offsets.marginLeft = marginLeft;

}

if (isIE10 ? parent.contains(scrollParent) : parent === scrollParent && scrollParent.nodeName !== 'BODY') {

offsets = includeScroll(offsets, parent);

}

return offsets;

}

function getViewportOffsetRectRelativeToArtbitraryNode(element) {

var html = element.ownerDocument.documentElement;

var relativeOffset = getOffsetRectRelativeToArbitraryNode(element, html);

var width = Math.max(html.clientWidth, window.innerWidth || 0);

var height = Math.max(html.clientHeight, window.innerHeight || 0);

var scrollTop = getScroll(html);

var scrollLeft = getScroll(html, 'left');

var offset = {

top: scrollTop - relativeOffset.top + relativeOffset.marginTop,

left: scrollLeft - relativeOffset.left + relativeOffset.marginLeft,

width: width,

height: height

};

return getClientRect(offset);

}

/\*\*

\* Check if the given element is fixed or is inside a fixed parent

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element

\* @argument {Element} customContainer

\* @returns {Boolean} answer to "isFixed?"

\*/

function isFixed(element) {

var nodeName = element.nodeName;

if (nodeName === 'BODY' || nodeName === 'HTML') {

return false;

}

if (getStyleComputedProperty(element, 'position') === 'fixed') {

return true;

}

return isFixed(getParentNode(element));

}

/\*\*

\* Computed the boundaries limits and return them

\* @method

\* @memberof Popper.Utils

\* @param {HTMLElement} popper

\* @param {HTMLElement} reference

\* @param {number} padding

\* @param {HTMLElement} boundariesElement - Element used to define the boundaries

\* @returns {Object} Coordinates of the boundaries

\*/

function getBoundaries(popper, reference, padding, boundariesElement) {

// NOTE: 1 DOM access here

var boundaries = { top: 0, left: 0 };

var offsetParent = findCommonOffsetParent(popper, reference);

// Handle viewport case

if (boundariesElement === 'viewport') {

boundaries = getViewportOffsetRectRelativeToArtbitraryNode(offsetParent);

} else {

// Handle other cases based on DOM element used as boundaries

var boundariesNode = void 0;

if (boundariesElement === 'scrollParent') {

boundariesNode = getScrollParent(getParentNode(reference));

if (boundariesNode.nodeName === 'BODY') {

boundariesNode = popper.ownerDocument.documentElement;

}

} else if (boundariesElement === 'window') {

boundariesNode = popper.ownerDocument.documentElement;

} else {

boundariesNode = boundariesElement;

}

var offsets = getOffsetRectRelativeToArbitraryNode(boundariesNode, offsetParent);

// In case of HTML, we need a different computation

if (boundariesNode.nodeName === 'HTML' && !isFixed(offsetParent)) {

var \_getWindowSizes = getWindowSizes(),

height = \_getWindowSizes.height,

width = \_getWindowSizes.width;

boundaries.top += offsets.top - offsets.marginTop;

boundaries.bottom = height + offsets.top;

boundaries.left += offsets.left - offsets.marginLeft;

boundaries.right = width + offsets.left;

} else {

// for all the other DOM elements, this one is good

boundaries = offsets;

}

}

// Add paddings

boundaries.left += padding;

boundaries.top += padding;

boundaries.right -= padding;

boundaries.bottom -= padding;

return boundaries;

}

function getArea(\_ref) {

var width = \_ref.width,

height = \_ref.height;

return width \* height;

}

/\*\*

\* Utility used to transform the `auto` placement to the placement with more

\* available space.

\* @method

\* @memberof Popper.Utils

\* @argument {Object} data - The data object generated by update method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function computeAutoPlacement(placement, refRect, popper, reference, boundariesElement) {

var padding = arguments.length > 5 && arguments[5] !== undefined ? arguments[5] : 0;

if (placement.indexOf('auto') === -1) {

return placement;

}

var boundaries = getBoundaries(popper, reference, padding, boundariesElement);

var rects = {

top: {

width: boundaries.width,

height: refRect.top - boundaries.top

},

right: {

width: boundaries.right - refRect.right,

height: boundaries.height

},

bottom: {

width: boundaries.width,

height: boundaries.bottom - refRect.bottom

},

left: {

width: refRect.left - boundaries.left,

height: boundaries.height

}

};

var sortedAreas = Object.keys(rects).map(function (key) {

return \_extends$1({

key: key

}, rects[key], {

area: getArea(rects[key])

});

}).sort(function (a, b) {

return b.area - a.area;

});

var filteredAreas = sortedAreas.filter(function (\_ref2) {

var width = \_ref2.width,

height = \_ref2.height;

return width >= popper.clientWidth && height >= popper.clientHeight;

});

var computedPlacement = filteredAreas.length > 0 ? filteredAreas[0].key : sortedAreas[0].key;

var variation = placement.split('-')[1];

return computedPlacement + (variation ? '-' + variation : '');

}

/\*\*

\* Get offsets to the reference element

\* @method

\* @memberof Popper.Utils

\* @param {Object} state

\* @param {Element} popper - the popper element

\* @param {Element} reference - the reference element (the popper will be relative to this)

\* @returns {Object} An object containing the offsets which will be applied to the popper

\*/

function getReferenceOffsets(state, popper, reference) {

var commonOffsetParent = findCommonOffsetParent(popper, reference);

return getOffsetRectRelativeToArbitraryNode(reference, commonOffsetParent);

}

/\*\*

\* Get the outer sizes of the given element (offset size + margins)

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element

\* @returns {Object} object containing width and height properties

\*/

function getOuterSizes(element) {

var styles = getComputedStyle(element);

var x = parseFloat(styles.marginTop) + parseFloat(styles.marginBottom);

var y = parseFloat(styles.marginLeft) + parseFloat(styles.marginRight);

var result = {

width: element.offsetWidth + y,

height: element.offsetHeight + x

};

return result;

}

/\*\*

\* Get the opposite placement of the given one

\* @method

\* @memberof Popper.Utils

\* @argument {String} placement

\* @returns {String} flipped placement

\*/

function getOppositePlacement(placement) {

var hash = { left: 'right', right: 'left', bottom: 'top', top: 'bottom' };

return placement.replace(/left|right|bottom|top/g, function (matched) {

return hash[matched];

});

}

/\*\*

\* Get offsets to the popper

\* @method

\* @memberof Popper.Utils

\* @param {Object} position - CSS position the Popper will get applied

\* @param {HTMLElement} popper - the popper element

\* @param {Object} referenceOffsets - the reference offsets (the popper will be relative to this)

\* @param {String} placement - one of the valid placement options

\* @returns {Object} popperOffsets - An object containing the offsets which will be applied to the popper

\*/

function getPopperOffsets(popper, referenceOffsets, placement) {

placement = placement.split('-')[0];

// Get popper node sizes

var popperRect = getOuterSizes(popper);

// Add position, width and height to our offsets object

var popperOffsets = {

width: popperRect.width,

height: popperRect.height

};

// depending by the popper placement we have to compute its offsets slightly differently

var isHoriz = ['right', 'left'].indexOf(placement) !== -1;

var mainSide = isHoriz ? 'top' : 'left';

var secondarySide = isHoriz ? 'left' : 'top';

var measurement = isHoriz ? 'height' : 'width';

var secondaryMeasurement = !isHoriz ? 'height' : 'width';

popperOffsets[mainSide] = referenceOffsets[mainSide] + referenceOffsets[measurement] / 2 - popperRect[measurement] / 2;

if (placement === secondarySide) {

popperOffsets[secondarySide] = referenceOffsets[secondarySide] - popperRect[secondaryMeasurement];

} else {

popperOffsets[secondarySide] = referenceOffsets[getOppositePlacement(secondarySide)];

}

return popperOffsets;

}

/\*\*

\* Mimics the `find` method of Array

\* @method

\* @memberof Popper.Utils

\* @argument {Array} arr

\* @argument prop

\* @argument value

\* @returns index or -1

\*/

function find(arr, check) {

// use native find if supported

if (Array.prototype.find) {

return arr.find(check);

}

// use `filter` to obtain the same behavior of `find`

return arr.filter(check)[0];

}

/\*\*

\* Return the index of the matching object

\* @method

\* @memberof Popper.Utils

\* @argument {Array} arr

\* @argument prop

\* @argument value

\* @returns index or -1

\*/

function findIndex(arr, prop, value) {

// use native findIndex if supported

if (Array.prototype.findIndex) {

return arr.findIndex(function (cur) {

return cur[prop] === value;

});

}

// use `find` + `indexOf` if `findIndex` isn't supported

var match = find(arr, function (obj) {

return obj[prop] === value;

});

return arr.indexOf(match);

}

/\*\*

\* Loop trough the list of modifiers and run them in order,

\* each of them will then edit the data object.

\* @method

\* @memberof Popper.Utils

\* @param {dataObject} data

\* @param {Array} modifiers

\* @param {String} ends - Optional modifier name used as stopper

\* @returns {dataObject}

\*/

function runModifiers(modifiers, data, ends) {

var modifiersToRun = ends === undefined ? modifiers : modifiers.slice(0, findIndex(modifiers, 'name', ends));

modifiersToRun.forEach(function (modifier) {

if (modifier['function']) {

// eslint-disable-line dot-notation

console.warn('`modifier.function` is deprecated, use `modifier.fn`!');

}

var fn = modifier['function'] || modifier.fn; // eslint-disable-line dot-notation

if (modifier.enabled && isFunction(fn)) {

// Add properties to offsets to make them a complete clientRect object

// we do this before each modifier to make sure the previous one doesn't

// mess with these values

data.offsets.popper = getClientRect(data.offsets.popper);

data.offsets.reference = getClientRect(data.offsets.reference);

data = fn(data, modifier);

}

});

return data;

}

/\*\*

\* Updates the position of the popper, computing the new offsets and applying

\* the new style.<br />

\* Prefer `scheduleUpdate` over `update` because of performance reasons.

\* @method

\* @memberof Popper

\*/

function update() {

// if popper is destroyed, don't perform any further update

if (this.state.isDestroyed) {

return;

}

var data = {

instance: this,

styles: {},

arrowStyles: {},

attributes: {},

flipped: false,

offsets: {}

};

// compute reference element offsets

data.offsets.reference = getReferenceOffsets(this.state, this.popper, this.reference);

// compute auto placement, store placement inside the data object,

// modifiers will be able to edit `placement` if needed

// and refer to originalPlacement to know the original value

data.placement = computeAutoPlacement(this.options.placement, data.offsets.reference, this.popper, this.reference, this.options.modifiers.flip.boundariesElement, this.options.modifiers.flip.padding);

// store the computed placement inside `originalPlacement`

data.originalPlacement = data.placement;

// compute the popper offsets

data.offsets.popper = getPopperOffsets(this.popper, data.offsets.reference, data.placement);

data.offsets.popper.position = 'absolute';

// run the modifiers

data = runModifiers(this.modifiers, data);

// the first `update` will call `onCreate` callback

// the other ones will call `onUpdate` callback

if (!this.state.isCreated) {

this.state.isCreated = true;

this.options.onCreate(data);

} else {

this.options.onUpdate(data);

}

}

/\*\*

\* Helper used to know if the given modifier is enabled.

\* @method

\* @memberof Popper.Utils

\* @returns {Boolean}

\*/

function isModifierEnabled(modifiers, modifierName) {

return modifiers.some(function (\_ref) {

var name = \_ref.name,

enabled = \_ref.enabled;

return enabled && name === modifierName;

});

}

/\*\*

\* Get the prefixed supported property name

\* @method

\* @memberof Popper.Utils

\* @argument {String} property (camelCase)

\* @returns {String} prefixed property (camelCase or PascalCase, depending on the vendor prefix)

\*/

function getSupportedPropertyName(property) {

var prefixes = [false, 'ms', 'Webkit', 'Moz', 'O'];

var upperProp = property.charAt(0).toUpperCase() + property.slice(1);

for (var i = 0; i < prefixes.length - 1; i++) {

var prefix = prefixes[i];

var toCheck = prefix ? '' + prefix + upperProp : property;

if (typeof document.body.style[toCheck] !== 'undefined') {

return toCheck;

}

}

return null;

}

/\*\*

\* Destroy the popper

\* @method

\* @memberof Popper

\*/

function destroy() {

this.state.isDestroyed = true;

// touch DOM only if `applyStyle` modifier is enabled

if (isModifierEnabled(this.modifiers, 'applyStyle')) {

this.popper.removeAttribute('x-placement');

this.popper.style.left = '';

this.popper.style.position = '';

this.popper.style.top = '';

this.popper.style[getSupportedPropertyName('transform')] = '';

}

this.disableEventListeners();

// remove the popper if user explicity asked for the deletion on destroy

// do not use `remove` because IE11 doesn't support it

if (this.options.removeOnDestroy) {

this.popper.parentNode.removeChild(this.popper);

}

return this;

}

/\*\*

\* Get the window associated with the element

\* @argument {Element} element

\* @returns {Window}

\*/

function getWindow(element) {

var ownerDocument = element.ownerDocument;

return ownerDocument ? ownerDocument.defaultView : window;

}

function attachToScrollParents(scrollParent, event, callback, scrollParents) {

var isBody = scrollParent.nodeName === 'BODY';

var target = isBody ? scrollParent.ownerDocument.defaultView : scrollParent;

target.addEventListener(event, callback, { passive: true });

if (!isBody) {

attachToScrollParents(getScrollParent(target.parentNode), event, callback, scrollParents);

}

scrollParents.push(target);

}

/\*\*

\* Setup needed event listeners used to update the popper position

\* @method

\* @memberof Popper.Utils

\* @private

\*/

function setupEventListeners(reference, options, state, updateBound) {

// Resize event listener on window

state.updateBound = updateBound;

getWindow(reference).addEventListener('resize', state.updateBound, { passive: true });

// Scroll event listener on scroll parents

var scrollElement = getScrollParent(reference);

attachToScrollParents(scrollElement, 'scroll', state.updateBound, state.scrollParents);

state.scrollElement = scrollElement;

state.eventsEnabled = true;

return state;

}

/\*\*

\* It will add resize/scroll events and start recalculating

\* position of the popper element when they are triggered.

\* @method

\* @memberof Popper

\*/

function enableEventListeners() {

if (!this.state.eventsEnabled) {

this.state = setupEventListeners(this.reference, this.options, this.state, this.scheduleUpdate);

}

}

/\*\*

\* Remove event listeners used to update the popper position

\* @method

\* @memberof Popper.Utils

\* @private

\*/

function removeEventListeners(reference, state) {

// Remove resize event listener on window

getWindow(reference).removeEventListener('resize', state.updateBound);

// Remove scroll event listener on scroll parents

state.scrollParents.forEach(function (target) {

target.removeEventListener('scroll', state.updateBound);

});

// Reset state

state.updateBound = null;

state.scrollParents = [];

state.scrollElement = null;

state.eventsEnabled = false;

return state;

}

/\*\*

\* It will remove resize/scroll events and won't recalculate popper position

\* when they are triggered. It also won't trigger onUpdate callback anymore,

\* unless you call `update` method manually.

\* @method

\* @memberof Popper

\*/

function disableEventListeners() {

if (this.state.eventsEnabled) {

cancelAnimationFrame(this.scheduleUpdate);

this.state = removeEventListeners(this.reference, this.state);

}

}

/\*\*

\* Tells if a given input is a number

\* @method

\* @memberof Popper.Utils

\* @param {\*} input to check

\* @return {Boolean}

\*/

function isNumeric(n) {

return n !== '' && !isNaN(parseFloat(n)) && isFinite(n);

}

/\*\*

\* Set the style to the given popper

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element - Element to apply the style to

\* @argument {Object} styles

\* Object with a list of properties and values which will be applied to the element

\*/

function setStyles(element, styles) {

Object.keys(styles).forEach(function (prop) {

var unit = '';

// add unit if the value is numeric and is one of the following

if (['width', 'height', 'top', 'right', 'bottom', 'left'].indexOf(prop) !== -1 && isNumeric(styles[prop])) {

unit = 'px';

}

element.style[prop] = styles[prop] + unit;

});

}

/\*\*

\* Set the attributes to the given popper

\* @method

\* @memberof Popper.Utils

\* @argument {Element} element - Element to apply the attributes to

\* @argument {Object} styles

\* Object with a list of properties and values which will be applied to the element

\*/

function setAttributes(element, attributes) {

Object.keys(attributes).forEach(function (prop) {

var value = attributes[prop];

if (value !== false) {

element.setAttribute(prop, attributes[prop]);

} else {

element.removeAttribute(prop);

}

});

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by `update` method

\* @argument {Object} data.styles - List of style properties - values to apply to popper element

\* @argument {Object} data.attributes - List of attribute properties - values to apply to popper element

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The same data object

\*/

function applyStyle(data) {

// any property present in `data.styles` will be applied to the popper,

// in this way we can make the 3rd party modifiers add custom styles to it

// Be aware, modifiers could override the properties defined in the previous

// lines of this modifier!

setStyles(data.instance.popper, data.styles);

// any property present in `data.attributes` will be applied to the popper,

// they will be set as HTML attributes of the element

setAttributes(data.instance.popper, data.attributes);

// if arrowElement is defined and arrowStyles has some properties

if (data.arrowElement && Object.keys(data.arrowStyles).length) {

setStyles(data.arrowElement, data.arrowStyles);

}

return data;

}

/\*\*

\* Set the x-placement attribute before everything else because it could be used

\* to add margins to the popper margins needs to be calculated to get the

\* correct popper offsets.

\* @method

\* @memberof Popper.modifiers

\* @param {HTMLElement} reference - The reference element used to position the popper

\* @param {HTMLElement} popper - The HTML element used as popper.

\* @param {Object} options - Popper.js options

\*/

function applyStyleOnLoad(reference, popper, options, modifierOptions, state) {

// compute reference element offsets

var referenceOffsets = getReferenceOffsets(state, popper, reference);

// compute auto placement, store placement inside the data object,

// modifiers will be able to edit `placement` if needed

// and refer to originalPlacement to know the original value

var placement = computeAutoPlacement(options.placement, referenceOffsets, popper, reference, options.modifiers.flip.boundariesElement, options.modifiers.flip.padding);

popper.setAttribute('x-placement', placement);

// Apply `position` to popper before anything else because

// without the position applied we can't guarantee correct computations

setStyles(popper, { position: 'absolute' });

return options;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by `update` method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function computeStyle(data, options) {

var x = options.x,

y = options.y;

var popper = data.offsets.popper;

// Remove this legacy support in Popper.js v2

var legacyGpuAccelerationOption = find(data.instance.modifiers, function (modifier) {

return modifier.name === 'applyStyle';

}).gpuAcceleration;

if (legacyGpuAccelerationOption !== undefined) {

console.warn('WARNING: `gpuAcceleration` option moved to `computeStyle` modifier and will not be supported in future versions of Popper.js!');

}

var gpuAcceleration = legacyGpuAccelerationOption !== undefined ? legacyGpuAccelerationOption : options.gpuAcceleration;

var offsetParent = getOffsetParent(data.instance.popper);

var offsetParentRect = getBoundingClientRect(offsetParent);

// Styles

var styles = {

position: popper.position

};

// floor sides to avoid blurry text

var offsets = {

left: Math.floor(popper.left),

top: Math.floor(popper.top),

bottom: Math.floor(popper.bottom),

right: Math.floor(popper.right)

};

var sideA = x === 'bottom' ? 'top' : 'bottom';

var sideB = y === 'right' ? 'left' : 'right';

// if gpuAcceleration is set to `true` and transform is supported,

// we use `translate3d` to apply the position to the popper we

// automatically use the supported prefixed version if needed

var prefixedProperty = getSupportedPropertyName('transform');

// now, let's make a step back and look at this code closely (wtf?)

// If the content of the popper grows once it's been positioned, it

// may happen that the popper gets misplaced because of the new content

// overflowing its reference element

// To avoid this problem, we provide two options (x and y), which allow

// the consumer to define the offset origin.

// If we position a popper on top of a reference element, we can set

// `x` to `top` to make the popper grow towards its top instead of

// its bottom.

var left = void 0,

top = void 0;

if (sideA === 'bottom') {

top = -offsetParentRect.height + offsets.bottom;

} else {

top = offsets.top;

}

if (sideB === 'right') {

left = -offsetParentRect.width + offsets.right;

} else {

left = offsets.left;

}

if (gpuAcceleration && prefixedProperty) {

styles[prefixedProperty] = 'translate3d(' + left + 'px, ' + top + 'px, 0)';

styles[sideA] = 0;

styles[sideB] = 0;

styles.willChange = 'transform';

} else {

// othwerise, we use the standard `top`, `left`, `bottom` and `right` properties

var invertTop = sideA === 'bottom' ? -1 : 1;

var invertLeft = sideB === 'right' ? -1 : 1;

styles[sideA] = top \* invertTop;

styles[sideB] = left \* invertLeft;

styles.willChange = sideA + ', ' + sideB;

}

// Attributes

var attributes = {

'x-placement': data.placement

};

// Update `data` attributes, styles and arrowStyles

data.attributes = \_extends$1({}, attributes, data.attributes);

data.styles = \_extends$1({}, styles, data.styles);

data.arrowStyles = \_extends$1({}, data.offsets.arrow, data.arrowStyles);

return data;

}

/\*\*

\* Helper used to know if the given modifier depends from another one.<br />

\* It checks if the needed modifier is listed and enabled.

\* @method

\* @memberof Popper.Utils

\* @param {Array} modifiers - list of modifiers

\* @param {String} requestingName - name of requesting modifier

\* @param {String} requestedName - name of requested modifier

\* @returns {Boolean}

\*/

function isModifierRequired(modifiers, requestingName, requestedName) {

var requesting = find(modifiers, function (\_ref) {

var name = \_ref.name;

return name === requestingName;

});

var isRequired = !!requesting && modifiers.some(function (modifier) {

return modifier.name === requestedName && modifier.enabled && modifier.order < requesting.order;

});

if (!isRequired) {

var \_requesting = '`' + requestingName + '`';

var requested = '`' + requestedName + '`';

console.warn(requested + ' modifier is required by ' + \_requesting + ' modifier in order to work, be sure to include it before ' + \_requesting + '!');

}

return isRequired;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by update method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function arrow(data, options) {

var \_data$offsets$arrow;

// arrow depends on keepTogether in order to work

if (!isModifierRequired(data.instance.modifiers, 'arrow', 'keepTogether')) {

return data;

}

var arrowElement = options.element;

// if arrowElement is a string, suppose it's a CSS selector

if (typeof arrowElement === 'string') {

arrowElement = data.instance.popper.querySelector(arrowElement);

// if arrowElement is not found, don't run the modifier

if (!arrowElement) {

return data;

}

} else {

// if the arrowElement isn't a query selector we must check that the

// provided DOM node is child of its popper node

if (!data.instance.popper.contains(arrowElement)) {

console.warn('WARNING: `arrow.element` must be child of its popper element!');

return data;

}

}

var placement = data.placement.split('-')[0];

var \_data$offsets = data.offsets,

popper = \_data$offsets.popper,

reference = \_data$offsets.reference;

var isVertical = ['left', 'right'].indexOf(placement) !== -1;

var len = isVertical ? 'height' : 'width';

var sideCapitalized = isVertical ? 'Top' : 'Left';

var side = sideCapitalized.toLowerCase();

var altSide = isVertical ? 'left' : 'top';

var opSide = isVertical ? 'bottom' : 'right';

var arrowElementSize = getOuterSizes(arrowElement)[len];

//

// extends keepTogether behavior making sure the popper and its

// reference have enough pixels in conjuction

//

// top/left side

if (reference[opSide] - arrowElementSize < popper[side]) {

data.offsets.popper[side] -= popper[side] - (reference[opSide] - arrowElementSize);

}

// bottom/right side

if (reference[side] + arrowElementSize > popper[opSide]) {

data.offsets.popper[side] += reference[side] + arrowElementSize - popper[opSide];

}

data.offsets.popper = getClientRect(data.offsets.popper);

// compute center of the popper

var center = reference[side] + reference[len] / 2 - arrowElementSize / 2;

// Compute the sideValue using the updated popper offsets

// take popper margin in account because we don't have this info available

var css = getStyleComputedProperty(data.instance.popper);

var popperMarginSide = parseFloat(css['margin' + sideCapitalized], 10);

var popperBorderSide = parseFloat(css['border' + sideCapitalized + 'Width'], 10);

var sideValue = center - data.offsets.popper[side] - popperMarginSide - popperBorderSide;

// prevent arrowElement from being placed not contiguously to its popper

sideValue = Math.max(Math.min(popper[len] - arrowElementSize, sideValue), 0);

data.arrowElement = arrowElement;

data.offsets.arrow = (\_data$offsets$arrow = {}, defineProperty(\_data$offsets$arrow, side, Math.round(sideValue)), defineProperty(\_data$offsets$arrow, altSide, ''), \_data$offsets$arrow);

return data;

}

/\*\*

\* Get the opposite placement variation of the given one

\* @method

\* @memberof Popper.Utils

\* @argument {String} placement variation

\* @returns {String} flipped placement variation

\*/

function getOppositeVariation(variation) {

if (variation === 'end') {

return 'start';

} else if (variation === 'start') {

return 'end';

}

return variation;

}

/\*\*

\* List of accepted placements to use as values of the `placement` option.<br />

\* Valid placements are:

\* - `auto`

\* - `top`

\* - `right`

\* - `bottom`

\* - `left`

\*

\* Each placement can have a variation from this list:

\* - `-start`

\* - `-end`

\*

\* Variations are interpreted easily if you think of them as the left to right

\* written languages. Horizontally (`top` and `bottom`), `start` is left and `end`

\* is right.<br />

\* Vertically (`left` and `right`), `start` is top and `end` is bottom.

\*

\* Some valid examples are:

\* - `top-end` (on top of reference, right aligned)

\* - `right-start` (on right of reference, top aligned)

\* - `bottom` (on bottom, centered)

\* - `auto-right` (on the side with more space available, alignment depends by placement)

\*

\* @static

\* @type {Array}

\* @enum {String}

\* @readonly

\* @method placements

\* @memberof Popper

\*/

var placements = ['auto-start', 'auto', 'auto-end', 'top-start', 'top', 'top-end', 'right-start', 'right', 'right-end', 'bottom-end', 'bottom', 'bottom-start', 'left-end', 'left', 'left-start'];

// Get rid of `auto` `auto-start` and `auto-end`

var validPlacements = placements.slice(3);

/\*\*

\* Given an initial placement, returns all the subsequent placements

\* clockwise (or counter-clockwise).

\*

\* @method

\* @memberof Popper.Utils

\* @argument {String} placement - A valid placement (it accepts variations)

\* @argument {Boolean} counter - Set to true to walk the placements counterclockwise

\* @returns {Array} placements including their variations

\*/

function clockwise(placement) {

var counter = arguments.length > 1 && arguments[1] !== undefined ? arguments[1] : false;

var index = validPlacements.indexOf(placement);

var arr = validPlacements.slice(index + 1).concat(validPlacements.slice(0, index));

return counter ? arr.reverse() : arr;

}

var BEHAVIORS = {

FLIP: 'flip',

CLOCKWISE: 'clockwise',

COUNTERCLOCKWISE: 'counterclockwise'

};

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by update method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function flip(data, options) {

// if `inner` modifier is enabled, we can't use the `flip` modifier

if (isModifierEnabled(data.instance.modifiers, 'inner')) {

return data;

}

if (data.flipped && data.placement === data.originalPlacement) {

// seems like flip is trying to loop, probably there's not enough space on any of the flippable sides

return data;

}

var boundaries = getBoundaries(data.instance.popper, data.instance.reference, options.padding, options.boundariesElement);

var placement = data.placement.split('-')[0];

var placementOpposite = getOppositePlacement(placement);

var variation = data.placement.split('-')[1] || '';

var flipOrder = [];

switch (options.behavior) {

case BEHAVIORS.FLIP:

flipOrder = [placement, placementOpposite];

break;

case BEHAVIORS.CLOCKWISE:

flipOrder = clockwise(placement);

break;

case BEHAVIORS.COUNTERCLOCKWISE:

flipOrder = clockwise(placement, true);

break;

default:

flipOrder = options.behavior;

}

flipOrder.forEach(function (step, index) {

if (placement !== step || flipOrder.length === index + 1) {

return data;

}

placement = data.placement.split('-')[0];

placementOpposite = getOppositePlacement(placement);

var popperOffsets = data.offsets.popper;

var refOffsets = data.offsets.reference;

// using floor because the reference offsets may contain decimals we are not going to consider here

var floor = Math.floor;

var overlapsRef = placement === 'left' && floor(popperOffsets.right) > floor(refOffsets.left) || placement === 'right' && floor(popperOffsets.left) < floor(refOffsets.right) || placement === 'top' && floor(popperOffsets.bottom) > floor(refOffsets.top) || placement === 'bottom' && floor(popperOffsets.top) < floor(refOffsets.bottom);

var overflowsLeft = floor(popperOffsets.left) < floor(boundaries.left);

var overflowsRight = floor(popperOffsets.right) > floor(boundaries.right);

var overflowsTop = floor(popperOffsets.top) < floor(boundaries.top);

var overflowsBottom = floor(popperOffsets.bottom) > floor(boundaries.bottom);

var overflowsBoundaries = placement === 'left' && overflowsLeft || placement === 'right' && overflowsRight || placement === 'top' && overflowsTop || placement === 'bottom' && overflowsBottom;

// flip the variation if required

var isVertical = ['top', 'bottom'].indexOf(placement) !== -1;

var flippedVariation = !!options.flipVariations && (isVertical && variation === 'start' && overflowsLeft || isVertical && variation === 'end' && overflowsRight || !isVertical && variation === 'start' && overflowsTop || !isVertical && variation === 'end' && overflowsBottom);

if (overlapsRef || overflowsBoundaries || flippedVariation) {

// this boolean to detect any flip loop

data.flipped = true;

if (overlapsRef || overflowsBoundaries) {

placement = flipOrder[index + 1];

}

if (flippedVariation) {

variation = getOppositeVariation(variation);

}

data.placement = placement + (variation ? '-' + variation : '');

// this object contains `position`, we want to preserve it along with

// any additional property we may add in the future

data.offsets.popper = \_extends$1({}, data.offsets.popper, getPopperOffsets(data.instance.popper, data.offsets.reference, data.placement));

data = runModifiers(data.instance.modifiers, data, 'flip');

}

});

return data;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by update method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function keepTogether(data) {

var \_data$offsets = data.offsets,

popper = \_data$offsets.popper,

reference = \_data$offsets.reference;

var placement = data.placement.split('-')[0];

var floor = Math.floor;

var isVertical = ['top', 'bottom'].indexOf(placement) !== -1;

var side = isVertical ? 'right' : 'bottom';

var opSide = isVertical ? 'left' : 'top';

var measurement = isVertical ? 'width' : 'height';

if (popper[side] < floor(reference[opSide])) {

data.offsets.popper[opSide] = floor(reference[opSide]) - popper[measurement];

}

if (popper[opSide] > floor(reference[side])) {

data.offsets.popper[opSide] = floor(reference[side]);

}

return data;

}

/\*\*

\* Converts a string containing value + unit into a px value number

\* @function

\* @memberof {modifiers~offset}

\* @private

\* @argument {String} str - Value + unit string

\* @argument {String} measurement - `height` or `width`

\* @argument {Object} popperOffsets

\* @argument {Object} referenceOffsets

\* @returns {Number|String}

\* Value in pixels, or original string if no values were extracted

\*/

function toValue(str, measurement, popperOffsets, referenceOffsets) {

// separate value from unit

var split = str.match(/((?:\-|\+)?\d\*\.?\d\*)(.\*)/);

var value = +split[1];

var unit = split[2];

// If it's not a number it's an operator, I guess

if (!value) {

return str;

}

if (unit.indexOf('%') === 0) {

var element = void 0;

switch (unit) {

case '%p':

element = popperOffsets;

break;

case '%':

case '%r':

default:

element = referenceOffsets;

}

var rect = getClientRect(element);

return rect[measurement] / 100 \* value;

} else if (unit === 'vh' || unit === 'vw') {

// if is a vh or vw, we calculate the size based on the viewport

var size = void 0;

if (unit === 'vh') {

size = Math.max(document.documentElement.clientHeight, window.innerHeight || 0);

} else {

size = Math.max(document.documentElement.clientWidth, window.innerWidth || 0);

}

return size / 100 \* value;

} else {

// if is an explicit pixel unit, we get rid of the unit and keep the value

// if is an implicit unit, it's px, and we return just the value

return value;

}

}

/\*\*

\* Parse an `offset` string to extrapolate `x` and `y` numeric offsets.

\* @function

\* @memberof {modifiers~offset}

\* @private

\* @argument {String} offset

\* @argument {Object} popperOffsets

\* @argument {Object} referenceOffsets

\* @argument {String} basePlacement

\* @returns {Array} a two cells array with x and y offsets in numbers

\*/

function parseOffset(offset, popperOffsets, referenceOffsets, basePlacement) {

var offsets = [0, 0];

// Use height if placement is left or right and index is 0 otherwise use width

// in this way the first offset will use an axis and the second one

// will use the other one

var useHeight = ['right', 'left'].indexOf(basePlacement) !== -1;

// Split the offset string to obtain a list of values and operands

// The regex addresses values with the plus or minus sign in front (+10, -20, etc)

var fragments = offset.split(/(\+|\-)/).map(function (frag) {

return frag.trim();

});

// Detect if the offset string contains a pair of values or a single one

// they could be separated by comma or space

var divider = fragments.indexOf(find(fragments, function (frag) {

return frag.search(/,|\s/) !== -1;

}));

if (fragments[divider] && fragments[divider].indexOf(',') === -1) {

console.warn('Offsets separated by white space(s) are deprecated, use a comma (,) instead.');

}

// If divider is found, we divide the list of values and operands to divide

// them by ofset X and Y.

var splitRegex = /\s\*,\s\*|\s+/;

var ops = divider !== -1 ? [fragments.slice(0, divider).concat([fragments[divider].split(splitRegex)[0]]), [fragments[divider].split(splitRegex)[1]].concat(fragments.slice(divider + 1))] : [fragments];

// Convert the values with units to absolute pixels to allow our computations

ops = ops.map(function (op, index) {

// Most of the units rely on the orientation of the popper

var measurement = (index === 1 ? !useHeight : useHeight) ? 'height' : 'width';

var mergeWithPrevious = false;

return op

// This aggregates any `+` or `-` sign that aren't considered operators

// e.g.: 10 + +5 => [10, +, +5]

.reduce(function (a, b) {

if (a[a.length - 1] === '' && ['+', '-'].indexOf(b) !== -1) {

a[a.length - 1] = b;

mergeWithPrevious = true;

return a;

} else if (mergeWithPrevious) {

a[a.length - 1] += b;

mergeWithPrevious = false;

return a;

} else {

return a.concat(b);

}

}, [])

// Here we convert the string values into number values (in px)

.map(function (str) {

return toValue(str, measurement, popperOffsets, referenceOffsets);

});

});

// Loop trough the offsets arrays and execute the operations

ops.forEach(function (op, index) {

op.forEach(function (frag, index2) {

if (isNumeric(frag)) {

offsets[index] += frag \* (op[index2 - 1] === '-' ? -1 : 1);

}

});

});

return offsets;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by update method

\* @argument {Object} options - Modifiers configuration and options

\* @argument {Number|String} options.offset=0

\* The offset value as described in the modifier description

\* @returns {Object} The data object, properly modified

\*/

function offset(data, \_ref) {

var offset = \_ref.offset;

var placement = data.placement,

\_data$offsets = data.offsets,

popper = \_data$offsets.popper,

reference = \_data$offsets.reference;

var basePlacement = placement.split('-')[0];

var offsets = void 0;

if (isNumeric(+offset)) {

offsets = [+offset, 0];

} else {

offsets = parseOffset(offset, popper, reference, basePlacement);

}

if (basePlacement === 'left') {

popper.top += offsets[0];

popper.left -= offsets[1];

} else if (basePlacement === 'right') {

popper.top += offsets[0];

popper.left += offsets[1];

} else if (basePlacement === 'top') {

popper.left += offsets[0];

popper.top -= offsets[1];

} else if (basePlacement === 'bottom') {

popper.left += offsets[0];

popper.top += offsets[1];

}

data.popper = popper;

return data;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by `update` method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function preventOverflow(data, options) {

var boundariesElement = options.boundariesElement || getOffsetParent(data.instance.popper);

// If offsetParent is the reference element, we really want to

// go one step up and use the next offsetParent as reference to

// avoid to make this modifier completely useless and look like broken

if (data.instance.reference === boundariesElement) {

boundariesElement = getOffsetParent(boundariesElement);

}

var boundaries = getBoundaries(data.instance.popper, data.instance.reference, options.padding, boundariesElement);

options.boundaries = boundaries;

var order = options.priority;

var popper = data.offsets.popper;

var check = {

primary: function primary(placement) {

var value = popper[placement];

if (popper[placement] < boundaries[placement] && !options.escapeWithReference) {

value = Math.max(popper[placement], boundaries[placement]);

}

return defineProperty({}, placement, value);

},

secondary: function secondary(placement) {

var mainSide = placement === 'right' ? 'left' : 'top';

var value = popper[mainSide];

if (popper[placement] > boundaries[placement] && !options.escapeWithReference) {

value = Math.min(popper[mainSide], boundaries[placement] - (placement === 'right' ? popper.width : popper.height));

}

return defineProperty({}, mainSide, value);

}

};

order.forEach(function (placement) {

var side = ['left', 'top'].indexOf(placement) !== -1 ? 'primary' : 'secondary';

popper = \_extends$1({}, popper, check[side](placement));

});

data.offsets.popper = popper;

return data;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by `update` method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function shift(data) {

var placement = data.placement;

var basePlacement = placement.split('-')[0];

var shiftvariation = placement.split('-')[1];

// if shift shiftvariation is specified, run the modifier

if (shiftvariation) {

var \_data$offsets = data.offsets,

reference = \_data$offsets.reference,

popper = \_data$offsets.popper;

var isVertical = ['bottom', 'top'].indexOf(basePlacement) !== -1;

var side = isVertical ? 'left' : 'top';

var measurement = isVertical ? 'width' : 'height';

var shiftOffsets = {

start: defineProperty({}, side, reference[side]),

end: defineProperty({}, side, reference[side] + reference[measurement] - popper[measurement])

};

data.offsets.popper = \_extends$1({}, popper, shiftOffsets[shiftvariation]);

}

return data;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by update method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function hide(data) {

if (!isModifierRequired(data.instance.modifiers, 'hide', 'preventOverflow')) {

return data;

}

var refRect = data.offsets.reference;

var bound = find(data.instance.modifiers, function (modifier) {

return modifier.name === 'preventOverflow';

}).boundaries;

if (refRect.bottom < bound.top || refRect.left > bound.right || refRect.top > bound.bottom || refRect.right < bound.left) {

// Avoid unnecessary DOM access if visibility hasn't changed

if (data.hide === true) {

return data;

}

data.hide = true;

data.attributes['x-out-of-boundaries'] = '';

} else {

// Avoid unnecessary DOM access if visibility hasn't changed

if (data.hide === false) {

return data;

}

data.hide = false;

data.attributes['x-out-of-boundaries'] = false;

}

return data;

}

/\*\*

\* @function

\* @memberof Modifiers

\* @argument {Object} data - The data object generated by `update` method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {Object} The data object, properly modified

\*/

function inner(data) {

var placement = data.placement;

var basePlacement = placement.split('-')[0];

var \_data$offsets = data.offsets,

popper = \_data$offsets.popper,

reference = \_data$offsets.reference;

var isHoriz = ['left', 'right'].indexOf(basePlacement) !== -1;

var subtractLength = ['top', 'left'].indexOf(basePlacement) === -1;

popper[isHoriz ? 'left' : 'top'] = reference[basePlacement] - (subtractLength ? popper[isHoriz ? 'width' : 'height'] : 0);

data.placement = getOppositePlacement(placement);

data.offsets.popper = getClientRect(popper);

return data;

}

/\*\*

\* Modifier function, each modifier can have a function of this type assigned

\* to its `fn` property.<br />

\* These functions will be called on each update, this means that you must

\* make sure they are performant enough to avoid performance bottlenecks.

\*

\* @function ModifierFn

\* @argument {dataObject} data - The data object generated by `update` method

\* @argument {Object} options - Modifiers configuration and options

\* @returns {dataObject} The data object, properly modified

\*/

/\*\*

\* Modifiers are plugins used to alter the behavior of your poppers.<br />

\* Popper.js uses a set of 9 modifiers to provide all the basic functionalities

\* needed by the library.

\*

\* Usually you don't want to override the `order`, `fn` and `onLoad` props.

\* All the other properties are configurations that could be tweaked.

\* @namespace modifiers

\*/

var modifiers = {

/\*\*

\* Modifier used to shift the popper on the start or end of its reference

\* element.<br />

\* It will read the variation of the `placement` property.<br />

\* It can be one either `-end` or `-start`.

\* @memberof modifiers

\* @inner

\*/

shift: {

/\*\* @prop {number} order=100 - Index used to define the order of execution \*/

order: 100,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: shift

},

/\*\*

\* The `offset` modifier can shift your popper on both its axis.

\*

\* It accepts the following units:

\* - `px` or unitless, interpreted as pixels

\* - `%` or `%r`, percentage relative to the length of the reference element

\* - `%p`, percentage relative to the length of the popper element

\* - `vw`, CSS viewport width unit

\* - `vh`, CSS viewport height unit

\*

\* For length is intended the main axis relative to the placement of the popper.<br />

\* This means that if the placement is `top` or `bottom`, the length will be the

\* `width`. In case of `left` or `right`, it will be the height.

\*

\* You can provide a single value (as `Number` or `String`), or a pair of values

\* as `String` divided by a comma or one (or more) white spaces.<br />

\* The latter is a deprecated method because it leads to confusion and will be

\* removed in v2.<br />

\* Additionally, it accepts additions and subtractions between different units.

\* Note that multiplications and divisions aren't supported.

\*

\* Valid examples are:

\* ```

\* 10

\* '10%'

\* '10, 10'

\* '10%, 10'

\* '10 + 10%'

\* '10 - 5vh + 3%'

\* '-10px + 5vh, 5px - 6%'

\* ```

\* > \*\*NB\*\*: If you desire to apply offsets to your poppers in a way that may make them overlap

\* > with their reference element, unfortunately, you will have to disable the `flip` modifier.

\* > More on this [reading this issue](https://github.com/FezVrasta/popper.js/issues/373)

\*

\* @memberof modifiers

\* @inner

\*/

offset: {

/\*\* @prop {number} order=200 - Index used to define the order of execution \*/

order: 200,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: offset,

/\*\* @prop {Number|String} offset=0

\* The offset value as described in the modifier description

\*/

offset: 0

},

/\*\*

\* Modifier used to prevent the popper from being positioned outside the boundary.

\*

\* An scenario exists where the reference itself is not within the boundaries.<br />

\* We can say it has "escaped the boundaries" — or just "escaped".<br />

\* In this case we need to decide whether the popper should either:

\*

\* - detach from the reference and remain "trapped" in the boundaries, or

\* - if it should ignore the boundary and "escape with its reference"

\*

\* When `escapeWithReference` is set to`true` and reference is completely

\* outside its boundaries, the popper will overflow (or completely leave)

\* the boundaries in order to remain attached to the edge of the reference.

\*

\* @memberof modifiers

\* @inner

\*/

preventOverflow: {

/\*\* @prop {number} order=300 - Index used to define the order of execution \*/

order: 300,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: preventOverflow,

/\*\*

\* @prop {Array} [priority=['left','right','top','bottom']]

\* Popper will try to prevent overflow following these priorities by default,

\* then, it could overflow on the left and on top of the `boundariesElement`

\*/

priority: ['left', 'right', 'top', 'bottom'],

/\*\*

\* @prop {number} padding=5

\* Amount of pixel used to define a minimum distance between the boundaries

\* and the popper this makes sure the popper has always a little padding

\* between the edges of its container

\*/

padding: 5,

/\*\*

\* @prop {String|HTMLElement} boundariesElement='scrollParent'

\* Boundaries used by the modifier, can be `scrollParent`, `window`,

\* `viewport` or any DOM element.

\*/

boundariesElement: 'scrollParent'

},

/\*\*

\* Modifier used to make sure the reference and its popper stay near eachothers

\* without leaving any gap between the two. Expecially useful when the arrow is

\* enabled and you want to assure it to point to its reference element.

\* It cares only about the first axis, you can still have poppers with margin

\* between the popper and its reference element.

\* @memberof modifiers

\* @inner

\*/

keepTogether: {

/\*\* @prop {number} order=400 - Index used to define the order of execution \*/

order: 400,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: keepTogether

},

/\*\*

\* This modifier is used to move the `arrowElement` of the popper to make

\* sure it is positioned between the reference element and its popper element.

\* It will read the outer size of the `arrowElement` node to detect how many

\* pixels of conjuction are needed.

\*

\* It has no effect if no `arrowElement` is provided.

\* @memberof modifiers

\* @inner

\*/

arrow: {

/\*\* @prop {number} order=500 - Index used to define the order of execution \*/

order: 500,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: arrow,

/\*\* @prop {String|HTMLElement} element='[x-arrow]' - Selector or node used as arrow \*/

element: '[x-arrow]'

},

/\*\*

\* Modifier used to flip the popper's placement when it starts to overlap its

\* reference element.

\*

\* Requires the `preventOverflow` modifier before it in order to work.

\*

\* \*\*NOTE:\*\* this modifier will interrupt the current update cycle and will

\* restart it if it detects the need to flip the placement.

\* @memberof modifiers

\* @inner

\*/

flip: {

/\*\* @prop {number} order=600 - Index used to define the order of execution \*/

order: 600,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: flip,

/\*\*

\* @prop {String|Array} behavior='flip'

\* The behavior used to change the popper's placement. It can be one of

\* `flip`, `clockwise`, `counterclockwise` or an array with a list of valid

\* placements (with optional variations).

\*/

behavior: 'flip',

/\*\*

\* @prop {number} padding=5

\* The popper will flip if it hits the edges of the `boundariesElement`

\*/

padding: 5,

/\*\*

\* @prop {String|HTMLElement} boundariesElement='viewport'

\* The element which will define the boundaries of the popper position,

\* the popper will never be placed outside of the defined boundaries

\* (except if keepTogether is enabled)

\*/

boundariesElement: 'viewport'

},

/\*\*

\* Modifier used to make the popper flow toward the inner of the reference element.

\* By default, when this modifier is disabled, the popper will be placed outside

\* the reference element.

\* @memberof modifiers

\* @inner

\*/

inner: {

/\*\* @prop {number} order=700 - Index used to define the order of execution \*/

order: 700,

/\*\* @prop {Boolean} enabled=false - Whether the modifier is enabled or not \*/

enabled: false,

/\*\* @prop {ModifierFn} \*/

fn: inner

},

/\*\*

\* Modifier used to hide the popper when its reference element is outside of the

\* popper boundaries. It will set a `x-out-of-boundaries` attribute which can

\* be used to hide with a CSS selector the popper when its reference is

\* out of boundaries.

\*

\* Requires the `preventOverflow` modifier before it in order to work.

\* @memberof modifiers

\* @inner

\*/

hide: {

/\*\* @prop {number} order=800 - Index used to define the order of execution \*/

order: 800,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: hide

},

/\*\*

\* Computes the style that will be applied to the popper element to gets

\* properly positioned.

\*

\* Note that this modifier will not touch the DOM, it just prepares the styles

\* so that `applyStyle` modifier can apply it. This separation is useful

\* in case you need to replace `applyStyle` with a custom implementation.

\*

\* This modifier has `850` as `order` value to maintain backward compatibility

\* with previous versions of Popper.js. Expect the modifiers ordering method

\* to change in future major versions of the library.

\*

\* @memberof modifiers

\* @inner

\*/

computeStyle: {

/\*\* @prop {number} order=850 - Index used to define the order of execution \*/

order: 850,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: computeStyle,

/\*\*

\* @prop {Boolean} gpuAcceleration=true

\* If true, it uses the CSS 3d transformation to position the popper.

\* Otherwise, it will use the `top` and `left` properties.

\*/

gpuAcceleration: true,

/\*\*

\* @prop {string} [x='bottom']

\* Where to anchor the X axis (`bottom` or `top`). AKA X offset origin.

\* Change this if your popper should grow in a direction different from `bottom`

\*/

x: 'bottom',

/\*\*

\* @prop {string} [x='left']

\* Where to anchor the Y axis (`left` or `right`). AKA Y offset origin.

\* Change this if your popper should grow in a direction different from `right`

\*/

y: 'right'

},

/\*\*

\* Applies the computed styles to the popper element.

\*

\* All the DOM manipulations are limited to this modifier. This is useful in case

\* you want to integrate Popper.js inside a framework or view library and you

\* want to delegate all the DOM manipulations to it.

\*

\* Note that if you disable this modifier, you must make sure the popper element

\* has its position set to `absolute` before Popper.js can do its work!

\*

\* Just disable this modifier and define you own to achieve the desired effect.

\*

\* @memberof modifiers

\* @inner

\*/

applyStyle: {

/\*\* @prop {number} order=900 - Index used to define the order of execution \*/

order: 900,

/\*\* @prop {Boolean} enabled=true - Whether the modifier is enabled or not \*/

enabled: true,

/\*\* @prop {ModifierFn} \*/

fn: applyStyle,

/\*\* @prop {Function} \*/

onLoad: applyStyleOnLoad,

/\*\*

\* @deprecated since version 1.10.0, the property moved to `computeStyle` modifier

\* @prop {Boolean} gpuAcceleration=true

\* If true, it uses the CSS 3d transformation to position the popper.

\* Otherwise, it will use the `top` and `left` properties.

\*/

gpuAcceleration: undefined

}

};

/\*\*

\* The `dataObject` is an object containing all the informations used by Popper.js

\* this object get passed to modifiers and to the `onCreate` and `onUpdate` callbacks.

\* @name dataObject

\* @property {Object} data.instance The Popper.js instance

\* @property {String} data.placement Placement applied to popper

\* @property {String} data.originalPlacement Placement originally defined on init

\* @property {Boolean} data.flipped True if popper has been flipped by flip modifier

\* @property {Boolean} data.hide True if the reference element is out of boundaries, useful to know when to hide the popper.

\* @property {HTMLElement} data.arrowElement Node used as arrow by arrow modifier

\* @property {Object} data.styles Any CSS property defined here will be applied to the popper, it expects the JavaScript nomenclature (eg. `marginBottom`)

\* @property {Object} data.arrowStyles Any CSS property defined here will be applied to the popper arrow, it expects the JavaScript nomenclature (eg. `marginBottom`)

\* @property {Object} data.boundaries Offsets of the popper boundaries

\* @property {Object} data.offsets The measurements of popper, reference and arrow elements.

\* @property {Object} data.offsets.popper `top`, `left`, `width`, `height` values

\* @property {Object} data.offsets.reference `top`, `left`, `width`, `height` values

\* @property {Object} data.offsets.arrow] `top` and `left` offsets, only one of them will be different from 0

\*/

/\*\*

\* Default options provided to Popper.js constructor.<br />

\* These can be overriden using the `options` argument of Popper.js.<br />

\* To override an option, simply pass as 3rd argument an object with the same

\* structure of this object, example:

\* ```

\* new Popper(ref, pop, {

\* modifiers: {

\* preventOverflow: { enabled: false }

\* }

\* })

\* ```

\* @type {Object}

\* @static

\* @memberof Popper

\*/

var Defaults = {

/\*\*

\* Popper's placement

\* @prop {Popper.placements} placement='bottom'

\*/

placement: 'bottom',

/\*\*

\* Whether events (resize, scroll) are initially enabled

\* @prop {Boolean} eventsEnabled=true

\*/

eventsEnabled: true,

/\*\*

\* Set to true if you want to automatically remove the popper when

\* you call the `destroy` method.

\* @prop {Boolean} removeOnDestroy=false

\*/

removeOnDestroy: false,

/\*\*

\* Callback called when the popper is created.<br />

\* By default, is set to no-op.<br />

\* Access Popper.js instance with `data.instance`.

\* @prop {onCreate}

\*/

onCreate: function onCreate() {},

/\*\*

\* Callback called when the popper is updated, this callback is not called

\* on the initialization/creation of the popper, but only on subsequent

\* updates.<br />

\* By default, is set to no-op.<br />

\* Access Popper.js instance with `data.instance`.

\* @prop {onUpdate}

\*/

onUpdate: function onUpdate() {},

/\*\*

\* List of modifiers used to modify the offsets before they are applied to the popper.

\* They provide most of the functionalities of Popper.js

\* @prop {modifiers}

\*/

modifiers: modifiers

};

/\*\*

\* @callback onCreate

\* @param {dataObject} data

\*/

/\*\*

\* @callback onUpdate

\* @param {dataObject} data

\*/

// Utils

// Methods

var Popper = function () {

/\*\*

\* Create a new Popper.js instance

\* @class Popper

\* @param {HTMLElement|referenceObject} reference - The reference element used to position the popper

\* @param {HTMLElement} popper - The HTML element used as popper.

\* @param {Object} options - Your custom options to override the ones defined in [Defaults](#defaults)

\* @return {Object} instance - The generated Popper.js instance

\*/

function Popper(reference, popper) {

var \_this = this;

var options = arguments.length > 2 && arguments[2] !== undefined ? arguments[2] : {};

classCallCheck(this, Popper);

this.scheduleUpdate = function () {

return requestAnimationFrame(\_this.update);

};

// make update() debounced, so that it only runs at most once-per-tick

this.update = debounce(this.update.bind(this));

// with {} we create a new object with the options inside it

this.options = \_extends$1({}, Popper.Defaults, options);

// init state

this.state = {

isDestroyed: false,

isCreated: false,

scrollParents: []

};

// get reference and popper elements (allow jQuery wrappers)

this.reference = reference && reference.jquery ? reference[0] : reference;

this.popper = popper && popper.jquery ? popper[0] : popper;

// Deep merge modifiers options

this.options.modifiers = {};

Object.keys(\_extends$1({}, Popper.Defaults.modifiers, options.modifiers)).forEach(function (name) {

\_this.options.modifiers[name] = \_extends$1({}, Popper.Defaults.modifiers[name] || {}, options.modifiers ? options.modifiers[name] : {});

});

// Refactoring modifiers' list (Object => Array)

this.modifiers = Object.keys(this.options.modifiers).map(function (name) {

return \_extends$1({

name: name

}, \_this.options.modifiers[name]);

})

// sort the modifiers by order

.sort(function (a, b) {

return a.order - b.order;

});

// modifiers have the ability to execute arbitrary code when Popper.js get inited

// such code is executed in the same order of its modifier

// they could add new properties to their options configuration

// BE AWARE: don't add options to `options.modifiers.name` but to `modifierOptions`!

this.modifiers.forEach(function (modifierOptions) {

if (modifierOptions.enabled && isFunction(modifierOptions.onLoad)) {

modifierOptions.onLoad(\_this.reference, \_this.popper, \_this.options, modifierOptions, \_this.state);

}

});

// fire the first update to position the popper in the right place

this.update();

var eventsEnabled = this.options.eventsEnabled;

if (eventsEnabled) {

// setup event listeners, they will take care of update the position in specific situations

this.enableEventListeners();

}

this.state.eventsEnabled = eventsEnabled;

}

// We can't use class properties because they don't get listed in the

// class prototype and break stuff like Sinon stubs

createClass(Popper, [{

key: 'update',

value: function update$$1() {

return update.call(this);

}

}, {

key: 'destroy',

value: function destroy$$1() {

return destroy.call(this);

}

}, {

key: 'enableEventListeners',

value: function enableEventListeners$$1() {

return enableEventListeners.call(this);

}

}, {

key: 'disableEventListeners',

value: function disableEventListeners$$1() {

return disableEventListeners.call(this);

}

/\*\*

\* Schedule an update, it will run on the next UI update available

\* @method scheduleUpdate

\* @memberof Popper

\*/

/\*\*

\* Collection of utilities useful when writing custom modifiers.

\* Starting from version 1.7, this method is available only if you

\* include `popper-utils.js` before `popper.js`.

\*

\* \*\*DEPRECATION\*\*: This way to access PopperUtils is deprecated

\* and will be removed in v2! Use the PopperUtils module directly instead.

\* Due to the high instability of the methods contained in Utils, we can't

\* guarantee them to follow semver. Use them at your own risk!

\* @static

\* @private

\* @type {Object}

\* @deprecated since version 1.8

\* @member Utils

\* @memberof Popper

\*/

}]);

return Popper;

}();

/\*\*

\* The `referenceObject` is an object that provides an interface compatible with Popper.js

\* and lets you use it as replacement of a real DOM node.<br />

\* You can use this method to position a popper relatively to a set of coordinates

\* in case you don't have a DOM node to use as reference.

\*

\* ```

\* new Popper(referenceObject, popperNode);

\* ```

\*

\* NB: This feature isn't supported in Internet Explorer 10

\* @name referenceObject

\* @property {Function} data.getBoundingClientRect

\* A function that returns a set of coordinates compatible with the native `getBoundingClientRect` method.

\* @property {number} data.clientWidth

\* An ES6 getter that will return the width of the virtual reference element.

\* @property {number} data.clientHeight

\* An ES6 getter that will return the height of the virtual reference element.

\*/

Popper.Utils = (typeof window !== 'undefined' ? window : global).PopperUtils;

Popper.placements = placements;

Popper.Defaults = Defaults;

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): dropdown.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Dropdown = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'dropdown';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.dropdown';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var ESCAPE\_KEYCODE = 27; // KeyboardEvent.which value for Escape (Esc) key

var SPACE\_KEYCODE = 32; // KeyboardEvent.which value for space key

var TAB\_KEYCODE = 9; // KeyboardEvent.which value for tab key

var ARROW\_UP\_KEYCODE = 38; // KeyboardEvent.which value for up arrow key

var ARROW\_DOWN\_KEYCODE = 40; // KeyboardEvent.which value for down arrow key

var RIGHT\_MOUSE\_BUTTON\_WHICH = 3; // MouseEvent.which value for the right button (assuming a right-handed mouse)

var REGEXP\_KEYDOWN = new RegExp(ARROW\_UP\_KEYCODE + "|" + ARROW\_DOWN\_KEYCODE + "|" + ESCAPE\_KEYCODE);

var Event = {

HIDE: "hide" + EVENT\_KEY,

HIDDEN: "hidden" + EVENT\_KEY,

SHOW: "show" + EVENT\_KEY,

SHOWN: "shown" + EVENT\_KEY,

CLICK: "click" + EVENT\_KEY,

CLICK\_DATA\_API: "click" + EVENT\_KEY + DATA\_API\_KEY,

KEYDOWN\_DATA\_API: "keydown" + EVENT\_KEY + DATA\_API\_KEY,

KEYUP\_DATA\_API: "keyup" + EVENT\_KEY + DATA\_API\_KEY

};

var ClassName = {

DISABLED: 'disabled',

SHOW: 'show',

DROPUP: 'dropup',

DROPRIGHT: 'dropright',

DROPLEFT: 'dropleft',

MENURIGHT: 'dropdown-menu-right',

MENULEFT: 'dropdown-menu-left',

POSITION\_STATIC: 'position-static'

};

var Selector = {

DATA\_TOGGLE: '[data-toggle="dropdown"]',

FORM\_CHILD: '.dropdown form',

MENU: '.dropdown-menu',

NAVBAR\_NAV: '.navbar-nav',

VISIBLE\_ITEMS: '.dropdown-menu .dropdown-item:not(.disabled)'

};

var AttachmentMap = {

TOP: 'top-start',

TOPEND: 'top-end',

BOTTOM: 'bottom-start',

BOTTOMEND: 'bottom-end',

RIGHT: 'right-start',

RIGHTEND: 'right-end',

LEFT: 'left-start',

LEFTEND: 'left-end'

};

var Default = {

offset: 0,

flip: true,

boundary: 'scrollParent'

};

var DefaultType = {

offset: '(number|string|function)',

flip: 'boolean',

boundary: '(string|element)'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Dropdown =

/\*#\_\_PURE\_\_\*/

function () {

function Dropdown(element, config) {

this.\_element = element;

this.\_popper = null;

this.\_config = this.\_getConfig(config);

this.\_menu = this.\_getMenuElement();

this.\_inNavbar = this.\_detectNavbar();

this.\_addEventListeners();

} // Getters

var \_proto = Dropdown.prototype;

// Public

\_proto.toggle = function toggle() {

if (this.\_element.disabled || $$$1(this.\_element).hasClass(ClassName.DISABLED)) {

return;

}

var parent = Dropdown.\_getParentFromElement(this.\_element);

var isActive = $$$1(this.\_menu).hasClass(ClassName.SHOW);

Dropdown.\_clearMenus();

if (isActive) {

return;

}

var relatedTarget = {

relatedTarget: this.\_element

};

var showEvent = $$$1.Event(Event.SHOW, relatedTarget);

$$$1(parent).trigger(showEvent);

if (showEvent.isDefaultPrevented()) {

return;

} // Disable totally Popper.js for Dropdown in Navbar

if (!this.\_inNavbar) {

/\*\*

\* Check for Popper dependency

\* Popper - https://popper.js.org

\*/

if (typeof Popper === 'undefined') {

throw new TypeError('Bootstrap dropdown require Popper.js (https://popper.js.org)');

}

var element = this.\_element; // For dropup with alignment we use the parent as popper container

if ($$$1(parent).hasClass(ClassName.DROPUP)) {

if ($$$1(this.\_menu).hasClass(ClassName.MENULEFT) || $$$1(this.\_menu).hasClass(ClassName.MENURIGHT)) {

element = parent;

}

} // If boundary is not `scrollParent`, then set position to `static`

// to allow the menu to "escape" the scroll parent's boundaries

// https://github.com/twbs/bootstrap/issues/24251

if (this.\_config.boundary !== 'scrollParent') {

$$$1(parent).addClass(ClassName.POSITION\_STATIC);

}

this.\_popper = new Popper(element, this.\_menu, this.\_getPopperConfig());

} // If this is a touch-enabled device we add extra

// empty mouseover listeners to the body's immediate children;

// only needed because of broken event delegation on iOS

// https://www.quirksmode.org/blog/archives/2014/02/mouse\_event\_bub.html

if ('ontouchstart' in document.documentElement && $$$1(parent).closest(Selector.NAVBAR\_NAV).length === 0) {

$$$1('body').children().on('mouseover', null, $$$1.noop);

}

this.\_element.focus();

this.\_element.setAttribute('aria-expanded', true);

$$$1(this.\_menu).toggleClass(ClassName.SHOW);

$$$1(parent).toggleClass(ClassName.SHOW).trigger($$$1.Event(Event.SHOWN, relatedTarget));

};

\_proto.dispose = function dispose() {

$$$1.removeData(this.\_element, DATA\_KEY);

$$$1(this.\_element).off(EVENT\_KEY);

this.\_element = null;

this.\_menu = null;

if (this.\_popper !== null) {

this.\_popper.destroy();

this.\_popper = null;

}

};

\_proto.update = function update() {

this.\_inNavbar = this.\_detectNavbar();

if (this.\_popper !== null) {

this.\_popper.scheduleUpdate();

}

}; // Private

\_proto.\_addEventListeners = function \_addEventListeners() {

var \_this = this;

$$$1(this.\_element).on(Event.CLICK, function (event) {

event.preventDefault();

event.stopPropagation();

\_this.toggle();

});

};

\_proto.\_getConfig = function \_getConfig(config) {

config = \_extends({}, this.constructor.Default, $$$1(this.\_element).data(), config);

Util.typeCheckConfig(NAME, config, this.constructor.DefaultType);

return config;

};

\_proto.\_getMenuElement = function \_getMenuElement() {

if (!this.\_menu) {

var parent = Dropdown.\_getParentFromElement(this.\_element);

this.\_menu = $$$1(parent).find(Selector.MENU)[0];

}

return this.\_menu;

};

\_proto.\_getPlacement = function \_getPlacement() {

var $parentDropdown = $$$1(this.\_element).parent();

var placement = AttachmentMap.BOTTOM; // Handle dropup

if ($parentDropdown.hasClass(ClassName.DROPUP)) {

placement = AttachmentMap.TOP;

if ($$$1(this.\_menu).hasClass(ClassName.MENURIGHT)) {

placement = AttachmentMap.TOPEND;

}

} else if ($parentDropdown.hasClass(ClassName.DROPRIGHT)) {

placement = AttachmentMap.RIGHT;

} else if ($parentDropdown.hasClass(ClassName.DROPLEFT)) {

placement = AttachmentMap.LEFT;

} else if ($$$1(this.\_menu).hasClass(ClassName.MENURIGHT)) {

placement = AttachmentMap.BOTTOMEND;

}

return placement;

};

\_proto.\_detectNavbar = function \_detectNavbar() {

return $$$1(this.\_element).closest('.navbar').length > 0;

};

\_proto.\_getPopperConfig = function \_getPopperConfig() {

var \_this2 = this;

var offsetConf = {};

if (typeof this.\_config.offset === 'function') {

offsetConf.fn = function (data) {

data.offsets = \_extends({}, data.offsets, \_this2.\_config.offset(data.offsets) || {});

return data;

};

} else {

offsetConf.offset = this.\_config.offset;

}

var popperConfig = {

placement: this.\_getPlacement(),

modifiers: {

offset: offsetConf,

flip: {

enabled: this.\_config.flip

},

preventOverflow: {

boundariesElement: this.\_config.boundary

}

}

};

return popperConfig;

}; // Static

Dropdown.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $$$1(this).data(DATA\_KEY);

var \_config = typeof config === 'object' ? config : null;

if (!data) {

data = new Dropdown(this, \_config);

$$$1(this).data(DATA\_KEY, data);

}

if (typeof config === 'string') {

if (typeof data[config] === 'undefined') {

throw new TypeError("No method named \"" + config + "\"");

}

data[config]();

}

});

};

Dropdown.\_clearMenus = function \_clearMenus(event) {

if (event && (event.which === RIGHT\_MOUSE\_BUTTON\_WHICH || event.type === 'keyup' && event.which !== TAB\_KEYCODE)) {

return;

}

var toggles = $$$1.makeArray($$$1(Selector.DATA\_TOGGLE));

for (var i = 0; i < toggles.length; i++) {

var parent = Dropdown.\_getParentFromElement(toggles[i]);

var context = $$$1(toggles[i]).data(DATA\_KEY);

var relatedTarget = {

relatedTarget: toggles[i]

};

if (!context) {

continue;

}

var dropdownMenu = context.\_menu;

if (!$$$1(parent).hasClass(ClassName.SHOW)) {

continue;

}

if (event && (event.type === 'click' && /input|textarea/i.test(event.target.tagName) || event.type === 'keyup' && event.which === TAB\_KEYCODE) && $$$1.contains(parent, event.target)) {

continue;

}

var hideEvent = $$$1.Event(Event.HIDE, relatedTarget);

$$$1(parent).trigger(hideEvent);

if (hideEvent.isDefaultPrevented()) {

continue;

} // If this is a touch-enabled device we remove the extra

// empty mouseover listeners we added for iOS support

if ('ontouchstart' in document.documentElement) {

$$$1('body').children().off('mouseover', null, $$$1.noop);

}

toggles[i].setAttribute('aria-expanded', 'false');

$$$1(dropdownMenu).removeClass(ClassName.SHOW);

$$$1(parent).removeClass(ClassName.SHOW).trigger($$$1.Event(Event.HIDDEN, relatedTarget));

}

};

Dropdown.\_getParentFromElement = function \_getParentFromElement(element) {

var parent;

var selector = Util.getSelectorFromElement(element);

if (selector) {

parent = $$$1(selector)[0];

}

return parent || element.parentNode;

}; // eslint-disable-next-line complexity

Dropdown.\_dataApiKeydownHandler = function \_dataApiKeydownHandler(event) {

// If not input/textarea:

// - And not a key in REGEXP\_KEYDOWN => not a dropdown command

// If input/textarea:

// - If space key => not a dropdown command

// - If key is other than escape

// - If key is not up or down => not a dropdown command

// - If trigger inside the menu => not a dropdown command

if (/input|textarea/i.test(event.target.tagName) ? event.which === SPACE\_KEYCODE || event.which !== ESCAPE\_KEYCODE && (event.which !== ARROW\_DOWN\_KEYCODE && event.which !== ARROW\_UP\_KEYCODE || $$$1(event.target).closest(Selector.MENU).length) : !REGEXP\_KEYDOWN.test(event.which)) {

return;

}

event.preventDefault();

event.stopPropagation();

if (this.disabled || $$$1(this).hasClass(ClassName.DISABLED)) {

return;

}

var parent = Dropdown.\_getParentFromElement(this);

var isActive = $$$1(parent).hasClass(ClassName.SHOW);

if (!isActive && (event.which !== ESCAPE\_KEYCODE || event.which !== SPACE\_KEYCODE) || isActive && (event.which === ESCAPE\_KEYCODE || event.which === SPACE\_KEYCODE)) {

if (event.which === ESCAPE\_KEYCODE) {

var toggle = $$$1(parent).find(Selector.DATA\_TOGGLE)[0];

$$$1(toggle).trigger('focus');

}

$$$1(this).trigger('click');

return;

}

var items = $$$1(parent).find(Selector.VISIBLE\_ITEMS).get();

if (items.length === 0) {

return;

}

var index = items.indexOf(event.target);

if (event.which === ARROW\_UP\_KEYCODE && index > 0) {

// Up

index--;

}

if (event.which === ARROW\_DOWN\_KEYCODE && index < items.length - 1) {

// Down

index++;

}

if (index < 0) {

index = 0;

}

items[index].focus();

};

\_createClass(Dropdown, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}, {

key: "Default",

get: function get() {

return Default;

}

}, {

key: "DefaultType",

get: function get() {

return DefaultType;

}

}]);

return Dropdown;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(document).on(Event.KEYDOWN\_DATA\_API, Selector.DATA\_TOGGLE, Dropdown.\_dataApiKeydownHandler).on(Event.KEYDOWN\_DATA\_API, Selector.MENU, Dropdown.\_dataApiKeydownHandler).on(Event.CLICK\_DATA\_API + " " + Event.KEYUP\_DATA\_API, Dropdown.\_clearMenus).on(Event.CLICK\_DATA\_API, Selector.DATA\_TOGGLE, function (event) {

event.preventDefault();

event.stopPropagation();

Dropdown.\_jQueryInterface.call($$$1(this), 'toggle');

}).on(Event.CLICK\_DATA\_API, Selector.FORM\_CHILD, function (e) {

e.stopPropagation();

});

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Dropdown.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Dropdown;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Dropdown.\_jQueryInterface;

};

return Dropdown;

}($, Popper);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): modal.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Modal = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'modal';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.modal';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var TRANSITION\_DURATION = 300;

var BACKDROP\_TRANSITION\_DURATION = 150;

var ESCAPE\_KEYCODE = 27; // KeyboardEvent.which value for Escape (Esc) key

var Default = {

backdrop: true,

keyboard: true,

focus: true,

show: true

};

var DefaultType = {

backdrop: '(boolean|string)',

keyboard: 'boolean',

focus: 'boolean',

show: 'boolean'

};

var Event = {

HIDE: "hide" + EVENT\_KEY,

HIDDEN: "hidden" + EVENT\_KEY,

SHOW: "show" + EVENT\_KEY,

SHOWN: "shown" + EVENT\_KEY,

FOCUSIN: "focusin" + EVENT\_KEY,

RESIZE: "resize" + EVENT\_KEY,

CLICK\_DISMISS: "click.dismiss" + EVENT\_KEY,

KEYDOWN\_DISMISS: "keydown.dismiss" + EVENT\_KEY,

MOUSEUP\_DISMISS: "mouseup.dismiss" + EVENT\_KEY,

MOUSEDOWN\_DISMISS: "mousedown.dismiss" + EVENT\_KEY,

CLICK\_DATA\_API: "click" + EVENT\_KEY + DATA\_API\_KEY

};

var ClassName = {

SCROLLBAR\_MEASURER: 'modal-scrollbar-measure',

BACKDROP: 'modal-backdrop',

OPEN: 'modal-open',

FADE: 'fade',

SHOW: 'show'

};

var Selector = {

DIALOG: '.modal-dialog',

DATA\_TOGGLE: '[data-toggle="modal"]',

DATA\_DISMISS: '[data-dismiss="modal"]',

FIXED\_CONTENT: '.fixed-top, .fixed-bottom, .is-fixed, .sticky-top',

STICKY\_CONTENT: '.sticky-top',

NAVBAR\_TOGGLER: '.navbar-toggler'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Modal =

/\*#\_\_PURE\_\_\*/

function () {

function Modal(element, config) {

this.\_config = this.\_getConfig(config);

this.\_element = element;

this.\_dialog = $$$1(element).find(Selector.DIALOG)[0];

this.\_backdrop = null;

this.\_isShown = false;

this.\_isBodyOverflowing = false;

this.\_ignoreBackdropClick = false;

this.\_originalBodyPadding = 0;

this.\_scrollbarWidth = 0;

} // Getters

var \_proto = Modal.prototype;

// Public

\_proto.toggle = function toggle(relatedTarget) {

return this.\_isShown ? this.hide() : this.show(relatedTarget);

};

\_proto.show = function show(relatedTarget) {

var \_this = this;

if (this.\_isTransitioning || this.\_isShown) {

return;

}

if (Util.supportsTransitionEnd() && $$$1(this.\_element).hasClass(ClassName.FADE)) {

this.\_isTransitioning = true;

}

var showEvent = $$$1.Event(Event.SHOW, {

relatedTarget: relatedTarget

});

$$$1(this.\_element).trigger(showEvent);

if (this.\_isShown || showEvent.isDefaultPrevented()) {

return;

}

this.\_isShown = true;

this.\_checkScrollbar();

this.\_setScrollbar();

this.\_adjustDialog();

$$$1(document.body).addClass(ClassName.OPEN);

this.\_setEscapeEvent();

this.\_setResizeEvent();

$$$1(this.\_element).on(Event.CLICK\_DISMISS, Selector.DATA\_DISMISS, function (event) {

return \_this.hide(event);

});

$$$1(this.\_dialog).on(Event.MOUSEDOWN\_DISMISS, function () {

$$$1(\_this.\_element).one(Event.MOUSEUP\_DISMISS, function (event) {

if ($$$1(event.target).is(\_this.\_element)) {

\_this.\_ignoreBackdropClick = true;

}

});

});

this.\_showBackdrop(function () {

return \_this.\_showElement(relatedTarget);

});

};

\_proto.hide = function hide(event) {

var \_this2 = this;

if (event) {

event.preventDefault();

}

if (this.\_isTransitioning || !this.\_isShown) {

return;

}

var hideEvent = $$$1.Event(Event.HIDE);

$$$1(this.\_element).trigger(hideEvent);

if (!this.\_isShown || hideEvent.isDefaultPrevented()) {

return;

}

this.\_isShown = false;

var transition = Util.supportsTransitionEnd() && $$$1(this.\_element).hasClass(ClassName.FADE);

if (transition) {

this.\_isTransitioning = true;

}

this.\_setEscapeEvent();

this.\_setResizeEvent();

$$$1(document).off(Event.FOCUSIN);

$$$1(this.\_element).removeClass(ClassName.SHOW);

$$$1(this.\_element).off(Event.CLICK\_DISMISS);

$$$1(this.\_dialog).off(Event.MOUSEDOWN\_DISMISS);

if (transition) {

$$$1(this.\_element).one(Util.TRANSITION\_END, function (event) {

return \_this2.\_hideModal(event);

}).emulateTransitionEnd(TRANSITION\_DURATION);

} else {

this.\_hideModal();

}

};

\_proto.dispose = function dispose() {

$$$1.removeData(this.\_element, DATA\_KEY);

$$$1(window, document, this.\_element, this.\_backdrop).off(EVENT\_KEY);

this.\_config = null;

this.\_element = null;

this.\_dialog = null;

this.\_backdrop = null;

this.\_isShown = null;

this.\_isBodyOverflowing = null;

this.\_ignoreBackdropClick = null;

this.\_scrollbarWidth = null;

};

\_proto.handleUpdate = function handleUpdate() {

this.\_adjustDialog();

}; // Private

\_proto.\_getConfig = function \_getConfig(config) {

config = \_extends({}, Default, config);

Util.typeCheckConfig(NAME, config, DefaultType);

return config;

};

\_proto.\_showElement = function \_showElement(relatedTarget) {

var \_this3 = this;

var transition = Util.supportsTransitionEnd() && $$$1(this.\_element).hasClass(ClassName.FADE);

if (!this.\_element.parentNode || this.\_element.parentNode.nodeType !== Node.ELEMENT\_NODE) {

// Don't move modal's DOM position

document.body.appendChild(this.\_element);

}

this.\_element.style.display = 'block';

this.\_element.removeAttribute('aria-hidden');

this.\_element.scrollTop = 0;

if (transition) {

Util.reflow(this.\_element);

}

$$$1(this.\_element).addClass(ClassName.SHOW);

if (this.\_config.focus) {

this.\_enforceFocus();

}

var shownEvent = $$$1.Event(Event.SHOWN, {

relatedTarget: relatedTarget

});

var transitionComplete = function transitionComplete() {

if (\_this3.\_config.focus) {

\_this3.\_element.focus();

}

\_this3.\_isTransitioning = false;

$$$1(\_this3.\_element).trigger(shownEvent);

};

if (transition) {

$$$1(this.\_dialog).one(Util.TRANSITION\_END, transitionComplete).emulateTransitionEnd(TRANSITION\_DURATION);

} else {

transitionComplete();

}

};

\_proto.\_enforceFocus = function \_enforceFocus() {

var \_this4 = this;

$$$1(document).off(Event.FOCUSIN) // Guard against infinite focus loop

.on(Event.FOCUSIN, function (event) {

if (document !== event.target && \_this4.\_element !== event.target && $$$1(\_this4.\_element).has(event.target).length === 0) {

\_this4.\_element.focus();

}

});

};

\_proto.\_setEscapeEvent = function \_setEscapeEvent() {

var \_this5 = this;

if (this.\_isShown && this.\_config.keyboard) {

$$$1(this.\_element).on(Event.KEYDOWN\_DISMISS, function (event) {

if (event.which === ESCAPE\_KEYCODE) {

event.preventDefault();

\_this5.hide();

}

});

} else if (!this.\_isShown) {

$$$1(this.\_element).off(Event.KEYDOWN\_DISMISS);

}

};

\_proto.\_setResizeEvent = function \_setResizeEvent() {

var \_this6 = this;

if (this.\_isShown) {

$$$1(window).on(Event.RESIZE, function (event) {

return \_this6.handleUpdate(event);

});

} else {

$$$1(window).off(Event.RESIZE);

}

};

\_proto.\_hideModal = function \_hideModal() {

var \_this7 = this;

this.\_element.style.display = 'none';

this.\_element.setAttribute('aria-hidden', true);

this.\_isTransitioning = false;

this.\_showBackdrop(function () {

$$$1(document.body).removeClass(ClassName.OPEN);

\_this7.\_resetAdjustments();

\_this7.\_resetScrollbar();

$$$1(\_this7.\_element).trigger(Event.HIDDEN);

});

};

\_proto.\_removeBackdrop = function \_removeBackdrop() {

if (this.\_backdrop) {

$$$1(this.\_backdrop).remove();

this.\_backdrop = null;

}

};

\_proto.\_showBackdrop = function \_showBackdrop(callback) {

var \_this8 = this;

var animate = $$$1(this.\_element).hasClass(ClassName.FADE) ? ClassName.FADE : '';

if (this.\_isShown && this.\_config.backdrop) {

var doAnimate = Util.supportsTransitionEnd() && animate;

this.\_backdrop = document.createElement('div');

this.\_backdrop.className = ClassName.BACKDROP;

if (animate) {

$$$1(this.\_backdrop).addClass(animate);

}

$$$1(this.\_backdrop).appendTo(document.body);

$$$1(this.\_element).on(Event.CLICK\_DISMISS, function (event) {

if (\_this8.\_ignoreBackdropClick) {

\_this8.\_ignoreBackdropClick = false;

return;

}

if (event.target !== event.currentTarget) {

return;

}

if (\_this8.\_config.backdrop === 'static') {

\_this8.\_element.focus();

} else {

\_this8.hide();

}

});

if (doAnimate) {

Util.reflow(this.\_backdrop);

}

$$$1(this.\_backdrop).addClass(ClassName.SHOW);

if (!callback) {

return;

}

if (!doAnimate) {

callback();

return;

}

$$$1(this.\_backdrop).one(Util.TRANSITION\_END, callback).emulateTransitionEnd(BACKDROP\_TRANSITION\_DURATION);

} else if (!this.\_isShown && this.\_backdrop) {

$$$1(this.\_backdrop).removeClass(ClassName.SHOW);

var callbackRemove = function callbackRemove() {

\_this8.\_removeBackdrop();

if (callback) {

callback();

}

};

if (Util.supportsTransitionEnd() && $$$1(this.\_element).hasClass(ClassName.FADE)) {

$$$1(this.\_backdrop).one(Util.TRANSITION\_END, callbackRemove).emulateTransitionEnd(BACKDROP\_TRANSITION\_DURATION);

} else {

callbackRemove();

}

} else if (callback) {

callback();

}

}; // ----------------------------------------------------------------------

// the following methods are used to handle overflowing modals

// todo (fat): these should probably be refactored out of modal.js

// ----------------------------------------------------------------------

\_proto.\_adjustDialog = function \_adjustDialog() {

var isModalOverflowing = this.\_element.scrollHeight > document.documentElement.clientHeight;

if (!this.\_isBodyOverflowing && isModalOverflowing) {

this.\_element.style.paddingLeft = this.\_scrollbarWidth + "px";

}

if (this.\_isBodyOverflowing && !isModalOverflowing) {

this.\_element.style.paddingRight = this.\_scrollbarWidth + "px";

}

};

\_proto.\_resetAdjustments = function \_resetAdjustments() {

this.\_element.style.paddingLeft = '';

this.\_element.style.paddingRight = '';

};

\_proto.\_checkScrollbar = function \_checkScrollbar() {

var rect = document.body.getBoundingClientRect();

this.\_isBodyOverflowing = rect.left + rect.right < window.innerWidth;

this.\_scrollbarWidth = this.\_getScrollbarWidth();

};

\_proto.\_setScrollbar = function \_setScrollbar() {

var \_this9 = this;

if (this.\_isBodyOverflowing) {

// Note: DOMNode.style.paddingRight returns the actual value or '' if not set

// while $(DOMNode).css('padding-right') returns the calculated value or 0 if not set

// Adjust fixed content padding

$$$1(Selector.FIXED\_CONTENT).each(function (index, element) {

var actualPadding = $$$1(element)[0].style.paddingRight;

var calculatedPadding = $$$1(element).css('padding-right');

$$$1(element).data('padding-right', actualPadding).css('padding-right', parseFloat(calculatedPadding) + \_this9.\_scrollbarWidth + "px");

}); // Adjust sticky content margin

$$$1(Selector.STICKY\_CONTENT).each(function (index, element) {

var actualMargin = $$$1(element)[0].style.marginRight;

var calculatedMargin = $$$1(element).css('margin-right');

$$$1(element).data('margin-right', actualMargin).css('margin-right', parseFloat(calculatedMargin) - \_this9.\_scrollbarWidth + "px");

}); // Adjust navbar-toggler margin

$$$1(Selector.NAVBAR\_TOGGLER).each(function (index, element) {

var actualMargin = $$$1(element)[0].style.marginRight;

var calculatedMargin = $$$1(element).css('margin-right');

$$$1(element).data('margin-right', actualMargin).css('margin-right', parseFloat(calculatedMargin) + \_this9.\_scrollbarWidth + "px");

}); // Adjust body padding

var actualPadding = document.body.style.paddingRight;

var calculatedPadding = $$$1('body').css('padding-right');

$$$1('body').data('padding-right', actualPadding).css('padding-right', parseFloat(calculatedPadding) + this.\_scrollbarWidth + "px");

}

};

\_proto.\_resetScrollbar = function \_resetScrollbar() {

// Restore fixed content padding

$$$1(Selector.FIXED\_CONTENT).each(function (index, element) {

var padding = $$$1(element).data('padding-right');

if (typeof padding !== 'undefined') {

$$$1(element).css('padding-right', padding).removeData('padding-right');

}

}); // Restore sticky content and navbar-toggler margin

$$$1(Selector.STICKY\_CONTENT + ", " + Selector.NAVBAR\_TOGGLER).each(function (index, element) {

var margin = $$$1(element).data('margin-right');

if (typeof margin !== 'undefined') {

$$$1(element).css('margin-right', margin).removeData('margin-right');

}

}); // Restore body padding

var padding = $$$1('body').data('padding-right');

if (typeof padding !== 'undefined') {

$$$1('body').css('padding-right', padding).removeData('padding-right');

}

};

\_proto.\_getScrollbarWidth = function \_getScrollbarWidth() {

// thx d.walsh

var scrollDiv = document.createElement('div');

scrollDiv.className = ClassName.SCROLLBAR\_MEASURER;

document.body.appendChild(scrollDiv);

var scrollbarWidth = scrollDiv.getBoundingClientRect().width - scrollDiv.clientWidth;

document.body.removeChild(scrollDiv);

return scrollbarWidth;

}; // Static

Modal.\_jQueryInterface = function \_jQueryInterface(config, relatedTarget) {

return this.each(function () {

var data = $$$1(this).data(DATA\_KEY);

var \_config = \_extends({}, Modal.Default, $$$1(this).data(), typeof config === 'object' && config);

if (!data) {

data = new Modal(this, \_config);

$$$1(this).data(DATA\_KEY, data);

}

if (typeof config === 'string') {

if (typeof data[config] === 'undefined') {

throw new TypeError("No method named \"" + config + "\"");

}

data[config](relatedTarget);

} else if (\_config.show) {

data.show(relatedTarget);

}

});

};

\_createClass(Modal, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}, {

key: "Default",

get: function get() {

return Default;

}

}]);

return Modal;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(document).on(Event.CLICK\_DATA\_API, Selector.DATA\_TOGGLE, function (event) {

var \_this10 = this;

var target;

var selector = Util.getSelectorFromElement(this);

if (selector) {

target = $$$1(selector)[0];

}

var config = $$$1(target).data(DATA\_KEY) ? 'toggle' : \_extends({}, $$$1(target).data(), $$$1(this).data());

if (this.tagName === 'A' || this.tagName === 'AREA') {

event.preventDefault();

}

var $target = $$$1(target).one(Event.SHOW, function (showEvent) {

if (showEvent.isDefaultPrevented()) {

// Only register focus restorer if modal will actually get shown

return;

}

$target.one(Event.HIDDEN, function () {

if ($$$1(\_this10).is(':visible')) {

\_this10.focus();

}

});

});

Modal.\_jQueryInterface.call($$$1(target), config, this);

});

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Modal.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Modal;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Modal.\_jQueryInterface;

};

return Modal;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): tooltip.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Tooltip = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'tooltip';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.tooltip';

var EVENT\_KEY = "." + DATA\_KEY;

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var TRANSITION\_DURATION = 150;

var CLASS\_PREFIX = 'bs-tooltip';

var BSCLS\_PREFIX\_REGEX = new RegExp("(^|\\s)" + CLASS\_PREFIX + "\\S+", 'g');

var DefaultType = {

animation: 'boolean',

template: 'string',

title: '(string|element|function)',

trigger: 'string',

delay: '(number|object)',

html: 'boolean',

selector: '(string|boolean)',

placement: '(string|function)',

offset: '(number|string)',

container: '(string|element|boolean)',

fallbackPlacement: '(string|array)',

boundary: '(string|element)'

};

var AttachmentMap = {

AUTO: 'auto',

TOP: 'top',

RIGHT: 'right',

BOTTOM: 'bottom',

LEFT: 'left'

};

var Default = {

animation: true,

template: '<div class="tooltip" role="tooltip">' + '<div class="arrow"></div>' + '<div class="tooltip-inner"></div></div>',

trigger: 'hover focus',

title: '',

delay: 0,

html: false,

selector: false,

placement: 'top',

offset: 0,

container: false,

fallbackPlacement: 'flip',

boundary: 'scrollParent'

};

var HoverState = {

SHOW: 'show',

OUT: 'out'

};

var Event = {

HIDE: "hide" + EVENT\_KEY,

HIDDEN: "hidden" + EVENT\_KEY,

SHOW: "show" + EVENT\_KEY,

SHOWN: "shown" + EVENT\_KEY,

INSERTED: "inserted" + EVENT\_KEY,

CLICK: "click" + EVENT\_KEY,

FOCUSIN: "focusin" + EVENT\_KEY,

FOCUSOUT: "focusout" + EVENT\_KEY,

MOUSEENTER: "mouseenter" + EVENT\_KEY,

MOUSELEAVE: "mouseleave" + EVENT\_KEY

};

var ClassName = {

FADE: 'fade',

SHOW: 'show'

};

var Selector = {

TOOLTIP: '.tooltip',

TOOLTIP\_INNER: '.tooltip-inner',

ARROW: '.arrow'

};

var Trigger = {

HOVER: 'hover',

FOCUS: 'focus',

CLICK: 'click',

MANUAL: 'manual'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Tooltip =

/\*#\_\_PURE\_\_\*/

function () {

function Tooltip(element, config) {

/\*\*

\* Check for Popper dependency

\* Popper - https://popper.js.org

\*/

if (typeof Popper === 'undefined') {

throw new TypeError('Bootstrap tooltips require Popper.js (https://popper.js.org)');

} // private

this.\_isEnabled = true;

this.\_timeout = 0;

this.\_hoverState = '';

this.\_activeTrigger = {};

this.\_popper = null; // Protected

this.element = element;

this.config = this.\_getConfig(config);

this.tip = null;

this.\_setListeners();

} // Getters

var \_proto = Tooltip.prototype;

// Public

\_proto.enable = function enable() {

this.\_isEnabled = true;

};

\_proto.disable = function disable() {

this.\_isEnabled = false;

};

\_proto.toggleEnabled = function toggleEnabled() {

this.\_isEnabled = !this.\_isEnabled;

};

\_proto.toggle = function toggle(event) {

if (!this.\_isEnabled) {

return;

}

if (event) {

var dataKey = this.constructor.DATA\_KEY;

var context = $$$1(event.currentTarget).data(dataKey);

if (!context) {

context = new this.constructor(event.currentTarget, this.\_getDelegateConfig());

$$$1(event.currentTarget).data(dataKey, context);

}

context.\_activeTrigger.click = !context.\_activeTrigger.click;

if (context.\_isWithActiveTrigger()) {

context.\_enter(null, context);

} else {

context.\_leave(null, context);

}

} else {

if ($$$1(this.getTipElement()).hasClass(ClassName.SHOW)) {

this.\_leave(null, this);

return;

}

this.\_enter(null, this);

}

};

\_proto.dispose = function dispose() {

clearTimeout(this.\_timeout);

$$$1.removeData(this.element, this.constructor.DATA\_KEY);

$$$1(this.element).off(this.constructor.EVENT\_KEY);

$$$1(this.element).closest('.modal').off('hide.bs.modal');

if (this.tip) {

$$$1(this.tip).remove();

}

this.\_isEnabled = null;

this.\_timeout = null;

this.\_hoverState = null;

this.\_activeTrigger = null;

if (this.\_popper !== null) {

this.\_popper.destroy();

}

this.\_popper = null;

this.element = null;

this.config = null;

this.tip = null;

};

\_proto.show = function show() {

var \_this = this;

if ($$$1(this.element).css('display') === 'none') {

throw new Error('Please use show on visible elements');

}

var showEvent = $$$1.Event(this.constructor.Event.SHOW);

if (this.isWithContent() && this.\_isEnabled) {

$$$1(this.element).trigger(showEvent);

var isInTheDom = $$$1.contains(this.element.ownerDocument.documentElement, this.element);

if (showEvent.isDefaultPrevented() || !isInTheDom) {

return;

}

var tip = this.getTipElement();

var tipId = Util.getUID(this.constructor.NAME);

tip.setAttribute('id', tipId);

this.element.setAttribute('aria-describedby', tipId);

this.setContent();

if (this.config.animation) {

$$$1(tip).addClass(ClassName.FADE);

}

var placement = typeof this.config.placement === 'function' ? this.config.placement.call(this, tip, this.element) : this.config.placement;

var attachment = this.\_getAttachment(placement);

this.addAttachmentClass(attachment);

var container = this.config.container === false ? document.body : $$$1(this.config.container);

$$$1(tip).data(this.constructor.DATA\_KEY, this);

if (!$$$1.contains(this.element.ownerDocument.documentElement, this.tip)) {

$$$1(tip).appendTo(container);

}

$$$1(this.element).trigger(this.constructor.Event.INSERTED);

this.\_popper = new Popper(this.element, tip, {

placement: attachment,

modifiers: {

offset: {

offset: this.config.offset

},

flip: {

behavior: this.config.fallbackPlacement

},

arrow: {

element: Selector.ARROW

},

preventOverflow: {

boundariesElement: this.config.boundary

}

},

onCreate: function onCreate(data) {

if (data.originalPlacement !== data.placement) {

\_this.\_handlePopperPlacementChange(data);

}

},

onUpdate: function onUpdate(data) {

\_this.\_handlePopperPlacementChange(data);

}

});

$$$1(tip).addClass(ClassName.SHOW); // If this is a touch-enabled device we add extra

// empty mouseover listeners to the body's immediate children;

// only needed because of broken event delegation on iOS

// https://www.quirksmode.org/blog/archives/2014/02/mouse\_event\_bub.html

if ('ontouchstart' in document.documentElement) {

$$$1('body').children().on('mouseover', null, $$$1.noop);

}

var complete = function complete() {

if (\_this.config.animation) {

\_this.\_fixTransition();

}

var prevHoverState = \_this.\_hoverState;

\_this.\_hoverState = null;

$$$1(\_this.element).trigger(\_this.constructor.Event.SHOWN);

if (prevHoverState === HoverState.OUT) {

\_this.\_leave(null, \_this);

}

};

if (Util.supportsTransitionEnd() && $$$1(this.tip).hasClass(ClassName.FADE)) {

$$$1(this.tip).one(Util.TRANSITION\_END, complete).emulateTransitionEnd(Tooltip.\_TRANSITION\_DURATION);

} else {

complete();

}

}

};

\_proto.hide = function hide(callback) {

var \_this2 = this;

var tip = this.getTipElement();

var hideEvent = $$$1.Event(this.constructor.Event.HIDE);

var complete = function complete() {

if (\_this2.\_hoverState !== HoverState.SHOW && tip.parentNode) {

tip.parentNode.removeChild(tip);

}

\_this2.\_cleanTipClass();

\_this2.element.removeAttribute('aria-describedby');

$$$1(\_this2.element).trigger(\_this2.constructor.Event.HIDDEN);

if (\_this2.\_popper !== null) {

\_this2.\_popper.destroy();

}

if (callback) {

callback();

}

};

$$$1(this.element).trigger(hideEvent);

if (hideEvent.isDefaultPrevented()) {

return;

}

$$$1(tip).removeClass(ClassName.SHOW); // If this is a touch-enabled device we remove the extra

// empty mouseover listeners we added for iOS support

if ('ontouchstart' in document.documentElement) {

$$$1('body').children().off('mouseover', null, $$$1.noop);

}

this.\_activeTrigger[Trigger.CLICK] = false;

this.\_activeTrigger[Trigger.FOCUS] = false;

this.\_activeTrigger[Trigger.HOVER] = false;

if (Util.supportsTransitionEnd() && $$$1(this.tip).hasClass(ClassName.FADE)) {

$$$1(tip).one(Util.TRANSITION\_END, complete).emulateTransitionEnd(TRANSITION\_DURATION);

} else {

complete();

}

this.\_hoverState = '';

};

\_proto.update = function update() {

if (this.\_popper !== null) {

this.\_popper.scheduleUpdate();

}

}; // Protected

\_proto.isWithContent = function isWithContent() {

return Boolean(this.getTitle());

};

\_proto.addAttachmentClass = function addAttachmentClass(attachment) {

$$$1(this.getTipElement()).addClass(CLASS\_PREFIX + "-" + attachment);

};

\_proto.getTipElement = function getTipElement() {

this.tip = this.tip || $$$1(this.config.template)[0];

return this.tip;

};

\_proto.setContent = function setContent() {

var $tip = $$$1(this.getTipElement());

this.setElementContent($tip.find(Selector.TOOLTIP\_INNER), this.getTitle());

$tip.removeClass(ClassName.FADE + " " + ClassName.SHOW);

};

\_proto.setElementContent = function setElementContent($element, content) {

var html = this.config.html;

if (typeof content === 'object' && (content.nodeType || content.jquery)) {

// Content is a DOM node or a jQuery

if (html) {

if (!$$$1(content).parent().is($element)) {

$element.empty().append(content);

}

} else {

$element.text($$$1(content).text());

}

} else {

$element[html ? 'html' : 'text'](content);

}

};

\_proto.getTitle = function getTitle() {

var title = this.element.getAttribute('data-original-title');

if (!title) {

title = typeof this.config.title === 'function' ? this.config.title.call(this.element) : this.config.title;

}

return title;

}; // Private

\_proto.\_getAttachment = function \_getAttachment(placement) {

return AttachmentMap[placement.toUpperCase()];

};

\_proto.\_setListeners = function \_setListeners() {

var \_this3 = this;

var triggers = this.config.trigger.split(' ');

triggers.forEach(function (trigger) {

if (trigger === 'click') {

$$$1(\_this3.element).on(\_this3.constructor.Event.CLICK, \_this3.config.selector, function (event) {

return \_this3.toggle(event);

});

} else if (trigger !== Trigger.MANUAL) {

var eventIn = trigger === Trigger.HOVER ? \_this3.constructor.Event.MOUSEENTER : \_this3.constructor.Event.FOCUSIN;

var eventOut = trigger === Trigger.HOVER ? \_this3.constructor.Event.MOUSELEAVE : \_this3.constructor.Event.FOCUSOUT;

$$$1(\_this3.element).on(eventIn, \_this3.config.selector, function (event) {

return \_this3.\_enter(event);

}).on(eventOut, \_this3.config.selector, function (event) {

return \_this3.\_leave(event);

});

}

$$$1(\_this3.element).closest('.modal').on('hide.bs.modal', function () {

return \_this3.hide();

});

});

if (this.config.selector) {

this.config = \_extends({}, this.config, {

trigger: 'manual',

selector: ''

});

} else {

this.\_fixTitle();

}

};

\_proto.\_fixTitle = function \_fixTitle() {

var titleType = typeof this.element.getAttribute('data-original-title');

if (this.element.getAttribute('title') || titleType !== 'string') {

this.element.setAttribute('data-original-title', this.element.getAttribute('title') || '');

this.element.setAttribute('title', '');

}

};

\_proto.\_enter = function \_enter(event, context) {

var dataKey = this.constructor.DATA\_KEY;

context = context || $$$1(event.currentTarget).data(dataKey);

if (!context) {

context = new this.constructor(event.currentTarget, this.\_getDelegateConfig());

$$$1(event.currentTarget).data(dataKey, context);

}

if (event) {

context.\_activeTrigger[event.type === 'focusin' ? Trigger.FOCUS : Trigger.HOVER] = true;

}

if ($$$1(context.getTipElement()).hasClass(ClassName.SHOW) || context.\_hoverState === HoverState.SHOW) {

context.\_hoverState = HoverState.SHOW;

return;

}

clearTimeout(context.\_timeout);

context.\_hoverState = HoverState.SHOW;

if (!context.config.delay || !context.config.delay.show) {

context.show();

return;

}

context.\_timeout = setTimeout(function () {

if (context.\_hoverState === HoverState.SHOW) {

context.show();

}

}, context.config.delay.show);

};

\_proto.\_leave = function \_leave(event, context) {

var dataKey = this.constructor.DATA\_KEY;

context = context || $$$1(event.currentTarget).data(dataKey);

if (!context) {

context = new this.constructor(event.currentTarget, this.\_getDelegateConfig());

$$$1(event.currentTarget).data(dataKey, context);

}

if (event) {

context.\_activeTrigger[event.type === 'focusout' ? Trigger.FOCUS : Trigger.HOVER] = false;

}

if (context.\_isWithActiveTrigger()) {

return;

}

clearTimeout(context.\_timeout);

context.\_hoverState = HoverState.OUT;

if (!context.config.delay || !context.config.delay.hide) {

context.hide();

return;

}

context.\_timeout = setTimeout(function () {

if (context.\_hoverState === HoverState.OUT) {

context.hide();

}

}, context.config.delay.hide);

};

\_proto.\_isWithActiveTrigger = function \_isWithActiveTrigger() {

for (var trigger in this.\_activeTrigger) {

if (this.\_activeTrigger[trigger]) {

return true;

}

}

return false;

};

\_proto.\_getConfig = function \_getConfig(config) {

config = \_extends({}, this.constructor.Default, $$$1(this.element).data(), config);

if (typeof config.delay === 'number') {

config.delay = {

show: config.delay,

hide: config.delay

};

}

if (typeof config.title === 'number') {

config.title = config.title.toString();

}

if (typeof config.content === 'number') {

config.content = config.content.toString();

}

Util.typeCheckConfig(NAME, config, this.constructor.DefaultType);

return config;

};

\_proto.\_getDelegateConfig = function \_getDelegateConfig() {

var config = {};

if (this.config) {

for (var key in this.config) {

if (this.constructor.Default[key] !== this.config[key]) {

config[key] = this.config[key];

}

}

}

return config;

};

\_proto.\_cleanTipClass = function \_cleanTipClass() {

var $tip = $$$1(this.getTipElement());

var tabClass = $tip.attr('class').match(BSCLS\_PREFIX\_REGEX);

if (tabClass !== null && tabClass.length > 0) {

$tip.removeClass(tabClass.join(''));

}

};

\_proto.\_handlePopperPlacementChange = function \_handlePopperPlacementChange(data) {

this.\_cleanTipClass();

this.addAttachmentClass(this.\_getAttachment(data.placement));

};

\_proto.\_fixTransition = function \_fixTransition() {

var tip = this.getTipElement();

var initConfigAnimation = this.config.animation;

if (tip.getAttribute('x-placement') !== null) {

return;

}

$$$1(tip).removeClass(ClassName.FADE);

this.config.animation = false;

this.hide();

this.show();

this.config.animation = initConfigAnimation;

}; // Static

Tooltip.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $$$1(this).data(DATA\_KEY);

var \_config = typeof config === 'object' && config;

if (!data && /dispose|hide/.test(config)) {

return;

}

if (!data) {

data = new Tooltip(this, \_config);

$$$1(this).data(DATA\_KEY, data);

}

if (typeof config === 'string') {

if (typeof data[config] === 'undefined') {

throw new TypeError("No method named \"" + config + "\"");

}

data[config]();

}

});

};

\_createClass(Tooltip, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}, {

key: "Default",

get: function get() {

return Default;

}

}, {

key: "NAME",

get: function get() {

return NAME;

}

}, {

key: "DATA\_KEY",

get: function get() {

return DATA\_KEY;

}

}, {

key: "Event",

get: function get() {

return Event;

}

}, {

key: "EVENT\_KEY",

get: function get() {

return EVENT\_KEY;

}

}, {

key: "DefaultType",

get: function get() {

return DefaultType;

}

}]);

return Tooltip;

}();

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Tooltip.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Tooltip;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Tooltip.\_jQueryInterface;

};

return Tooltip;

}($, Popper);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): popover.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Popover = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'popover';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.popover';

var EVENT\_KEY = "." + DATA\_KEY;

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var CLASS\_PREFIX = 'bs-popover';

var BSCLS\_PREFIX\_REGEX = new RegExp("(^|\\s)" + CLASS\_PREFIX + "\\S+", 'g');

var Default = \_extends({}, Tooltip.Default, {

placement: 'right',

trigger: 'click',

content: '',

template: '<div class="popover" role="tooltip">' + '<div class="arrow"></div>' + '<h3 class="popover-header"></h3>' + '<div class="popover-body"></div></div>'

});

var DefaultType = \_extends({}, Tooltip.DefaultType, {

content: '(string|element|function)'

});

var ClassName = {

FADE: 'fade',

SHOW: 'show'

};

var Selector = {

TITLE: '.popover-header',

CONTENT: '.popover-body'

};

var Event = {

HIDE: "hide" + EVENT\_KEY,

HIDDEN: "hidden" + EVENT\_KEY,

SHOW: "show" + EVENT\_KEY,

SHOWN: "shown" + EVENT\_KEY,

INSERTED: "inserted" + EVENT\_KEY,

CLICK: "click" + EVENT\_KEY,

FOCUSIN: "focusin" + EVENT\_KEY,

FOCUSOUT: "focusout" + EVENT\_KEY,

MOUSEENTER: "mouseenter" + EVENT\_KEY,

MOUSELEAVE: "mouseleave" + EVENT\_KEY

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Popover =

/\*#\_\_PURE\_\_\*/

function (\_Tooltip) {

\_inheritsLoose(Popover, \_Tooltip);

function Popover() {

return \_Tooltip.apply(this, arguments) || this;

}

var \_proto = Popover.prototype;

// Overrides

\_proto.isWithContent = function isWithContent() {

return this.getTitle() || this.\_getContent();

};

\_proto.addAttachmentClass = function addAttachmentClass(attachment) {

$$$1(this.getTipElement()).addClass(CLASS\_PREFIX + "-" + attachment);

};

\_proto.getTipElement = function getTipElement() {

this.tip = this.tip || $$$1(this.config.template)[0];

return this.tip;

};

\_proto.setContent = function setContent() {

var $tip = $$$1(this.getTipElement()); // We use append for html objects to maintain js events

this.setElementContent($tip.find(Selector.TITLE), this.getTitle());

var content = this.\_getContent();

if (typeof content === 'function') {

content = content.call(this.element);

}

this.setElementContent($tip.find(Selector.CONTENT), content);

$tip.removeClass(ClassName.FADE + " " + ClassName.SHOW);

}; // Private

\_proto.\_getContent = function \_getContent() {

return this.element.getAttribute('data-content') || this.config.content;

};

\_proto.\_cleanTipClass = function \_cleanTipClass() {

var $tip = $$$1(this.getTipElement());

var tabClass = $tip.attr('class').match(BSCLS\_PREFIX\_REGEX);

if (tabClass !== null && tabClass.length > 0) {

$tip.removeClass(tabClass.join(''));

}

}; // Static

Popover.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $$$1(this).data(DATA\_KEY);

var \_config = typeof config === 'object' ? config : null;

if (!data && /destroy|hide/.test(config)) {

return;

}

if (!data) {

data = new Popover(this, \_config);

$$$1(this).data(DATA\_KEY, data);

}

if (typeof config === 'string') {

if (typeof data[config] === 'undefined') {

throw new TypeError("No method named \"" + config + "\"");

}

data[config]();

}

});

};

\_createClass(Popover, null, [{

key: "VERSION",

// Getters

get: function get() {

return VERSION;

}

}, {

key: "Default",

get: function get() {

return Default;

}

}, {

key: "NAME",

get: function get() {

return NAME;

}

}, {

key: "DATA\_KEY",

get: function get() {

return DATA\_KEY;

}

}, {

key: "Event",

get: function get() {

return Event;

}

}, {

key: "EVENT\_KEY",

get: function get() {

return EVENT\_KEY;

}

}, {

key: "DefaultType",

get: function get() {

return DefaultType;

}

}]);

return Popover;

}(Tooltip);

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Popover.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Popover;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Popover.\_jQueryInterface;

};

return Popover;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): scrollspy.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var ScrollSpy = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'scrollspy';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.scrollspy';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var Default = {

offset: 10,

method: 'auto',

target: ''

};

var DefaultType = {

offset: 'number',

method: 'string',

target: '(string|element)'

};

var Event = {

ACTIVATE: "activate" + EVENT\_KEY,

SCROLL: "scroll" + EVENT\_KEY,

LOAD\_DATA\_API: "load" + EVENT\_KEY + DATA\_API\_KEY

};

var ClassName = {

DROPDOWN\_ITEM: 'dropdown-item',

DROPDOWN\_MENU: 'dropdown-menu',

ACTIVE: 'active'

};

var Selector = {

DATA\_SPY: '[data-spy="scroll"]',

ACTIVE: '.active',

NAV\_LIST\_GROUP: '.nav, .list-group',

NAV\_LINKS: '.nav-link',

NAV\_ITEMS: '.nav-item',

LIST\_ITEMS: '.list-group-item',

DROPDOWN: '.dropdown',

DROPDOWN\_ITEMS: '.dropdown-item',

DROPDOWN\_TOGGLE: '.dropdown-toggle'

};

var OffsetMethod = {

OFFSET: 'offset',

POSITION: 'position'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var ScrollSpy =

/\*#\_\_PURE\_\_\*/

function () {

function ScrollSpy(element, config) {

var \_this = this;

this.\_element = element;

this.\_scrollElement = element.tagName === 'BODY' ? window : element;

this.\_config = this.\_getConfig(config);

this.\_selector = this.\_config.target + " " + Selector.NAV\_LINKS + "," + (this.\_config.target + " " + Selector.LIST\_ITEMS + ",") + (this.\_config.target + " " + Selector.DROPDOWN\_ITEMS);

this.\_offsets = [];

this.\_targets = [];

this.\_activeTarget = null;

this.\_scrollHeight = 0;

$$$1(this.\_scrollElement).on(Event.SCROLL, function (event) {

return \_this.\_process(event);

});

this.refresh();

this.\_process();

} // Getters

var \_proto = ScrollSpy.prototype;

// Public

\_proto.refresh = function refresh() {

var \_this2 = this;

var autoMethod = this.\_scrollElement === this.\_scrollElement.window ? OffsetMethod.OFFSET : OffsetMethod.POSITION;

var offsetMethod = this.\_config.method === 'auto' ? autoMethod : this.\_config.method;

var offsetBase = offsetMethod === OffsetMethod.POSITION ? this.\_getScrollTop() : 0;

this.\_offsets = [];

this.\_targets = [];

this.\_scrollHeight = this.\_getScrollHeight();

var targets = $$$1.makeArray($$$1(this.\_selector));

targets.map(function (element) {

var target;

var targetSelector = Util.getSelectorFromElement(element);

if (targetSelector) {

target = $$$1(targetSelector)[0];

}

if (target) {

var targetBCR = target.getBoundingClientRect();

if (targetBCR.width || targetBCR.height) {

// TODO (fat): remove sketch reliance on jQuery position/offset

return [$$$1(target)[offsetMethod]().top + offsetBase, targetSelector];

}

}

return null;

}).filter(function (item) {

return item;

}).sort(function (a, b) {

return a[0] - b[0];

}).forEach(function (item) {

\_this2.\_offsets.push(item[0]);

\_this2.\_targets.push(item[1]);

});

};

\_proto.dispose = function dispose() {

$$$1.removeData(this.\_element, DATA\_KEY);

$$$1(this.\_scrollElement).off(EVENT\_KEY);

this.\_element = null;

this.\_scrollElement = null;

this.\_config = null;

this.\_selector = null;

this.\_offsets = null;

this.\_targets = null;

this.\_activeTarget = null;

this.\_scrollHeight = null;

}; // Private

\_proto.\_getConfig = function \_getConfig(config) {

config = \_extends({}, Default, config);

if (typeof config.target !== 'string') {

var id = $$$1(config.target).attr('id');

if (!id) {

id = Util.getUID(NAME);

$$$1(config.target).attr('id', id);

}

config.target = "#" + id;

}

Util.typeCheckConfig(NAME, config, DefaultType);

return config;

};

\_proto.\_getScrollTop = function \_getScrollTop() {

return this.\_scrollElement === window ? this.\_scrollElement.pageYOffset : this.\_scrollElement.scrollTop;

};

\_proto.\_getScrollHeight = function \_getScrollHeight() {

return this.\_scrollElement.scrollHeight || Math.max(document.body.scrollHeight, document.documentElement.scrollHeight);

};

\_proto.\_getOffsetHeight = function \_getOffsetHeight() {

return this.\_scrollElement === window ? window.innerHeight : this.\_scrollElement.getBoundingClientRect().height;

};

\_proto.\_process = function \_process() {

var scrollTop = this.\_getScrollTop() + this.\_config.offset;

var scrollHeight = this.\_getScrollHeight();

var maxScroll = this.\_config.offset + scrollHeight - this.\_getOffsetHeight();

if (this.\_scrollHeight !== scrollHeight) {

this.refresh();

}

if (scrollTop >= maxScroll) {

var target = this.\_targets[this.\_targets.length - 1];

if (this.\_activeTarget !== target) {

this.\_activate(target);

}

return;

}

if (this.\_activeTarget && scrollTop < this.\_offsets[0] && this.\_offsets[0] > 0) {

this.\_activeTarget = null;

this.\_clear();

return;

}

for (var i = this.\_offsets.length; i--;) {

var isActiveTarget = this.\_activeTarget !== this.\_targets[i] && scrollTop >= this.\_offsets[i] && (typeof this.\_offsets[i + 1] === 'undefined' || scrollTop < this.\_offsets[i + 1]);

if (isActiveTarget) {

this.\_activate(this.\_targets[i]);

}

}

};

\_proto.\_activate = function \_activate(target) {

this.\_activeTarget = target;

this.\_clear();

var queries = this.\_selector.split(','); // eslint-disable-next-line arrow-body-style

queries = queries.map(function (selector) {

return selector + "[data-target=\"" + target + "\"]," + (selector + "[href=\"" + target + "\"]");

});

var $link = $$$1(queries.join(','));

if ($link.hasClass(ClassName.DROPDOWN\_ITEM)) {

$link.closest(Selector.DROPDOWN).find(Selector.DROPDOWN\_TOGGLE).addClass(ClassName.ACTIVE);

$link.addClass(ClassName.ACTIVE);

} else {

// Set triggered link as active

$link.addClass(ClassName.ACTIVE); // Set triggered links parents as active

// With both <ul> and <nav> markup a parent is the previous sibling of any nav ancestor

$link.parents(Selector.NAV\_LIST\_GROUP).prev(Selector.NAV\_LINKS + ", " + Selector.LIST\_ITEMS).addClass(ClassName.ACTIVE); // Handle special case when .nav-link is inside .nav-item

$link.parents(Selector.NAV\_LIST\_GROUP).prev(Selector.NAV\_ITEMS).children(Selector.NAV\_LINKS).addClass(ClassName.ACTIVE);

}

$$$1(this.\_scrollElement).trigger(Event.ACTIVATE, {

relatedTarget: target

});

};

\_proto.\_clear = function \_clear() {

$$$1(this.\_selector).filter(Selector.ACTIVE).removeClass(ClassName.ACTIVE);

}; // Static

ScrollSpy.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var data = $$$1(this).data(DATA\_KEY);

var \_config = typeof config === 'object' && config;

if (!data) {

data = new ScrollSpy(this, \_config);

$$$1(this).data(DATA\_KEY, data);

}

if (typeof config === 'string') {

if (typeof data[config] === 'undefined') {

throw new TypeError("No method named \"" + config + "\"");

}

data[config]();

}

});

};

\_createClass(ScrollSpy, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}, {

key: "Default",

get: function get() {

return Default;

}

}]);

return ScrollSpy;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(window).on(Event.LOAD\_DATA\_API, function () {

var scrollSpys = $$$1.makeArray($$$1(Selector.DATA\_SPY));

for (var i = scrollSpys.length; i--;) {

var $spy = $$$1(scrollSpys[i]);

ScrollSpy.\_jQueryInterface.call($spy, $spy.data());

}

});

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = ScrollSpy.\_jQueryInterface;

$$$1.fn[NAME].Constructor = ScrollSpy;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return ScrollSpy.\_jQueryInterface;

};

return ScrollSpy;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0): tab.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

var Tab = function ($$$1) {

/\*\*

\* ------------------------------------------------------------------------

\* Constants

\* ------------------------------------------------------------------------

\*/

var NAME = 'tab';

var VERSION = '4.0.0';

var DATA\_KEY = 'bs.tab';

var EVENT\_KEY = "." + DATA\_KEY;

var DATA\_API\_KEY = '.data-api';

var JQUERY\_NO\_CONFLICT = $$$1.fn[NAME];

var TRANSITION\_DURATION = 150;

var Event = {

HIDE: "hide" + EVENT\_KEY,

HIDDEN: "hidden" + EVENT\_KEY,

SHOW: "show" + EVENT\_KEY,

SHOWN: "shown" + EVENT\_KEY,

CLICK\_DATA\_API: "click" + EVENT\_KEY + DATA\_API\_KEY

};

var ClassName = {

DROPDOWN\_MENU: 'dropdown-menu',

ACTIVE: 'active',

DISABLED: 'disabled',

FADE: 'fade',

SHOW: 'show'

};

var Selector = {

DROPDOWN: '.dropdown',

NAV\_LIST\_GROUP: '.nav, .list-group',

ACTIVE: '.active',

ACTIVE\_UL: '> li > .active',

DATA\_TOGGLE: '[data-toggle="tab"], [data-toggle="pill"], [data-toggle="list"]',

DROPDOWN\_TOGGLE: '.dropdown-toggle',

DROPDOWN\_ACTIVE\_CHILD: '> .dropdown-menu .active'

/\*\*

\* ------------------------------------------------------------------------

\* Class Definition

\* ------------------------------------------------------------------------

\*/

};

var Tab =

/\*#\_\_PURE\_\_\*/

function () {

function Tab(element) {

this.\_element = element;

} // Getters

var \_proto = Tab.prototype;

// Public

\_proto.show = function show() {

var \_this = this;

if (this.\_element.parentNode && this.\_element.parentNode.nodeType === Node.ELEMENT\_NODE && $$$1(this.\_element).hasClass(ClassName.ACTIVE) || $$$1(this.\_element).hasClass(ClassName.DISABLED)) {

return;

}

var target;

var previous;

var listElement = $$$1(this.\_element).closest(Selector.NAV\_LIST\_GROUP)[0];

var selector = Util.getSelectorFromElement(this.\_element);

if (listElement) {

var itemSelector = listElement.nodeName === 'UL' ? Selector.ACTIVE\_UL : Selector.ACTIVE;

previous = $$$1.makeArray($$$1(listElement).find(itemSelector));

previous = previous[previous.length - 1];

}

var hideEvent = $$$1.Event(Event.HIDE, {

relatedTarget: this.\_element

});

var showEvent = $$$1.Event(Event.SHOW, {

relatedTarget: previous

});

if (previous) {

$$$1(previous).trigger(hideEvent);

}

$$$1(this.\_element).trigger(showEvent);

if (showEvent.isDefaultPrevented() || hideEvent.isDefaultPrevented()) {

return;

}

if (selector) {

target = $$$1(selector)[0];

}

this.\_activate(this.\_element, listElement);

var complete = function complete() {

var hiddenEvent = $$$1.Event(Event.HIDDEN, {

relatedTarget: \_this.\_element

});

var shownEvent = $$$1.Event(Event.SHOWN, {

relatedTarget: previous

});

$$$1(previous).trigger(hiddenEvent);

$$$1(\_this.\_element).trigger(shownEvent);

};

if (target) {

this.\_activate(target, target.parentNode, complete);

} else {

complete();

}

};

\_proto.dispose = function dispose() {

$$$1.removeData(this.\_element, DATA\_KEY);

this.\_element = null;

}; // Private

\_proto.\_activate = function \_activate(element, container, callback) {

var \_this2 = this;

var activeElements;

if (container.nodeName === 'UL') {

activeElements = $$$1(container).find(Selector.ACTIVE\_UL);

} else {

activeElements = $$$1(container).children(Selector.ACTIVE);

}

var active = activeElements[0];

var isTransitioning = callback && Util.supportsTransitionEnd() && active && $$$1(active).hasClass(ClassName.FADE);

var complete = function complete() {

return \_this2.\_transitionComplete(element, active, callback);

};

if (active && isTransitioning) {

$$$1(active).one(Util.TRANSITION\_END, complete).emulateTransitionEnd(TRANSITION\_DURATION);

} else {

complete();

}

};

\_proto.\_transitionComplete = function \_transitionComplete(element, active, callback) {

if (active) {

$$$1(active).removeClass(ClassName.SHOW + " " + ClassName.ACTIVE);

var dropdownChild = $$$1(active.parentNode).find(Selector.DROPDOWN\_ACTIVE\_CHILD)[0];

if (dropdownChild) {

$$$1(dropdownChild).removeClass(ClassName.ACTIVE);

}

if (active.getAttribute('role') === 'tab') {

active.setAttribute('aria-selected', false);

}

}

$$$1(element).addClass(ClassName.ACTIVE);

if (element.getAttribute('role') === 'tab') {

element.setAttribute('aria-selected', true);

}

Util.reflow(element);

$$$1(element).addClass(ClassName.SHOW);

if (element.parentNode && $$$1(element.parentNode).hasClass(ClassName.DROPDOWN\_MENU)) {

var dropdownElement = $$$1(element).closest(Selector.DROPDOWN)[0];

if (dropdownElement) {

$$$1(dropdownElement).find(Selector.DROPDOWN\_TOGGLE).addClass(ClassName.ACTIVE);

}

element.setAttribute('aria-expanded', true);

}

if (callback) {

callback();

}

}; // Static

Tab.\_jQueryInterface = function \_jQueryInterface(config) {

return this.each(function () {

var $this = $$$1(this);

var data = $this.data(DATA\_KEY);

if (!data) {

data = new Tab(this);

$this.data(DATA\_KEY, data);

}

if (typeof config === 'string') {

if (typeof data[config] === 'undefined') {

throw new TypeError("No method named \"" + config + "\"");

}

data[config]();

}

});

};

\_createClass(Tab, null, [{

key: "VERSION",

get: function get() {

return VERSION;

}

}]);

return Tab;

}();

/\*\*

\* ------------------------------------------------------------------------

\* Data Api implementation

\* ------------------------------------------------------------------------

\*/

$$$1(document).on(Event.CLICK\_DATA\_API, Selector.DATA\_TOGGLE, function (event) {

event.preventDefault();

Tab.\_jQueryInterface.call($$$1(this), 'show');

});

/\*\*

\* ------------------------------------------------------------------------

\* jQuery

\* ------------------------------------------------------------------------

\*/

$$$1.fn[NAME] = Tab.\_jQueryInterface;

$$$1.fn[NAME].Constructor = Tab;

$$$1.fn[NAME].noConflict = function () {

$$$1.fn[NAME] = JQUERY\_NO\_CONFLICT;

return Tab.\_jQueryInterface;

};

return Tab;

}($);

/\*\*

\* --------------------------------------------------------------------------

\* Bootstrap (v4.0.0-alpha.6): index.js

\* Licensed under MIT (https://github.com/twbs/bootstrap/blob/master/LICENSE)

\* --------------------------------------------------------------------------

\*/

(function ($$$1) {

if (typeof $$$1 === 'undefined') {

throw new TypeError('Bootstrap\'s JavaScript requires jQuery. jQuery must be included before Bootstrap\'s JavaScript.');

}

var version = $$$1.fn.jquery.split(' ')[0].split('.');

var minMajor = 1;

var ltMajor = 2;

var minMinor = 9;

var minPatch = 1;

var maxMajor = 4;

if (version[0] < ltMajor && version[1] < minMinor || version[0] === minMajor && version[1] === minMinor && version[2] < minPatch || version[0] >= maxMajor) {

throw new Error('Bootstrap\'s JavaScript requires at least jQuery v1.9.1 but less than v4.0.0');

}

})($);

exports.Util = Util;

exports.Alert = Alert;

exports.Button = Button;

exports.Carousel = Carousel;

exports.Collapse = Collapse;

exports.Dropdown = Dropdown;

exports.Modal = Modal;

exports.Popover = Popover;

exports.Scrollspy = ScrollSpy;

exports.Tab = Tab;

exports.Tooltip = Tooltip;

Object.defineProperty(exports, '\_\_esModule', { value: true });

})));

//# sourceMappingURL=bootstrap.bundle.js.map