ion-select

SHADOW

Selects are form controls to select an option, or options, from a set of options, similar to a native `<select>` element. When a user taps the select, a dialog appears with all of the options in a large, easy to select list.

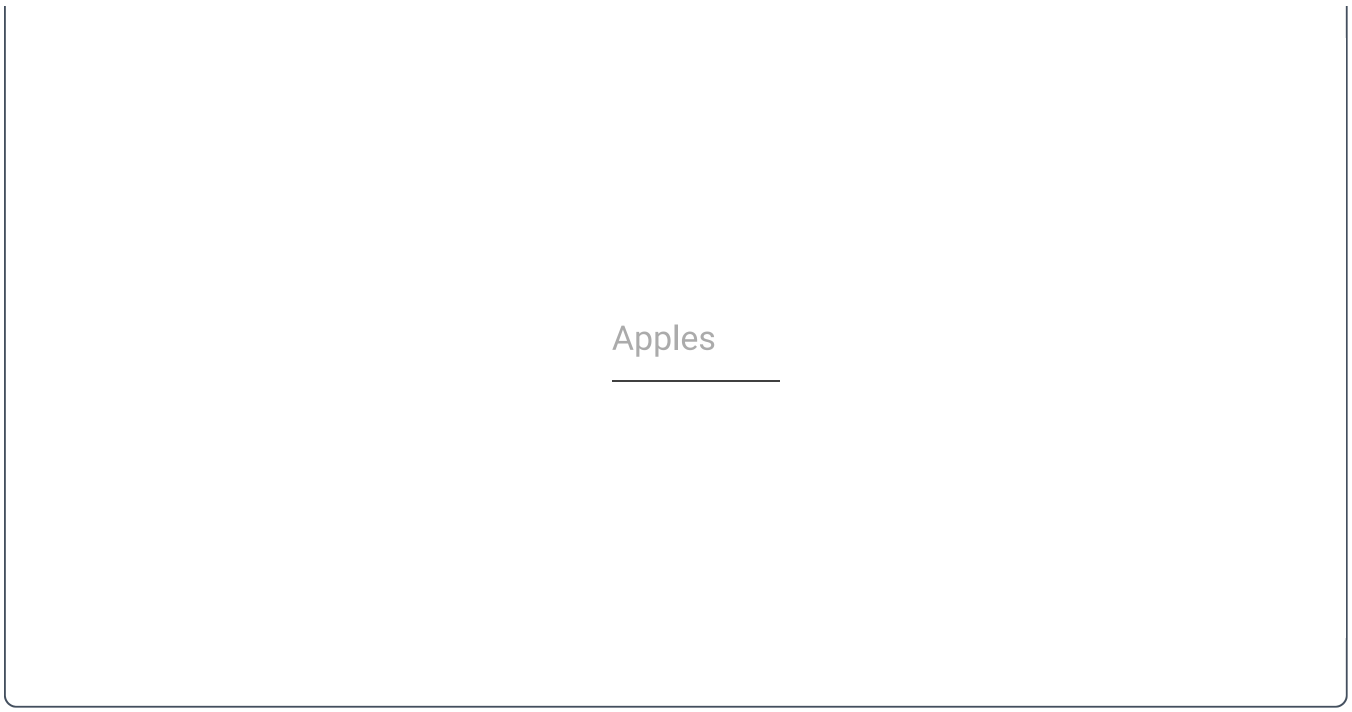
A select should be used with child `<ion-select-option>` elements. If the child option is not given a `value` attribute then its text will be used as the value.

If `value` is set on the `<ion-select>`, the selected option will be chosen based on that value.

Single Selection

By default, the select allows the user to select only one option. The alert interface presents users with a radio button styled list of options. The select component's value receives the value of the selected option's value.

Angular JavaScript React Vue | iOS MD

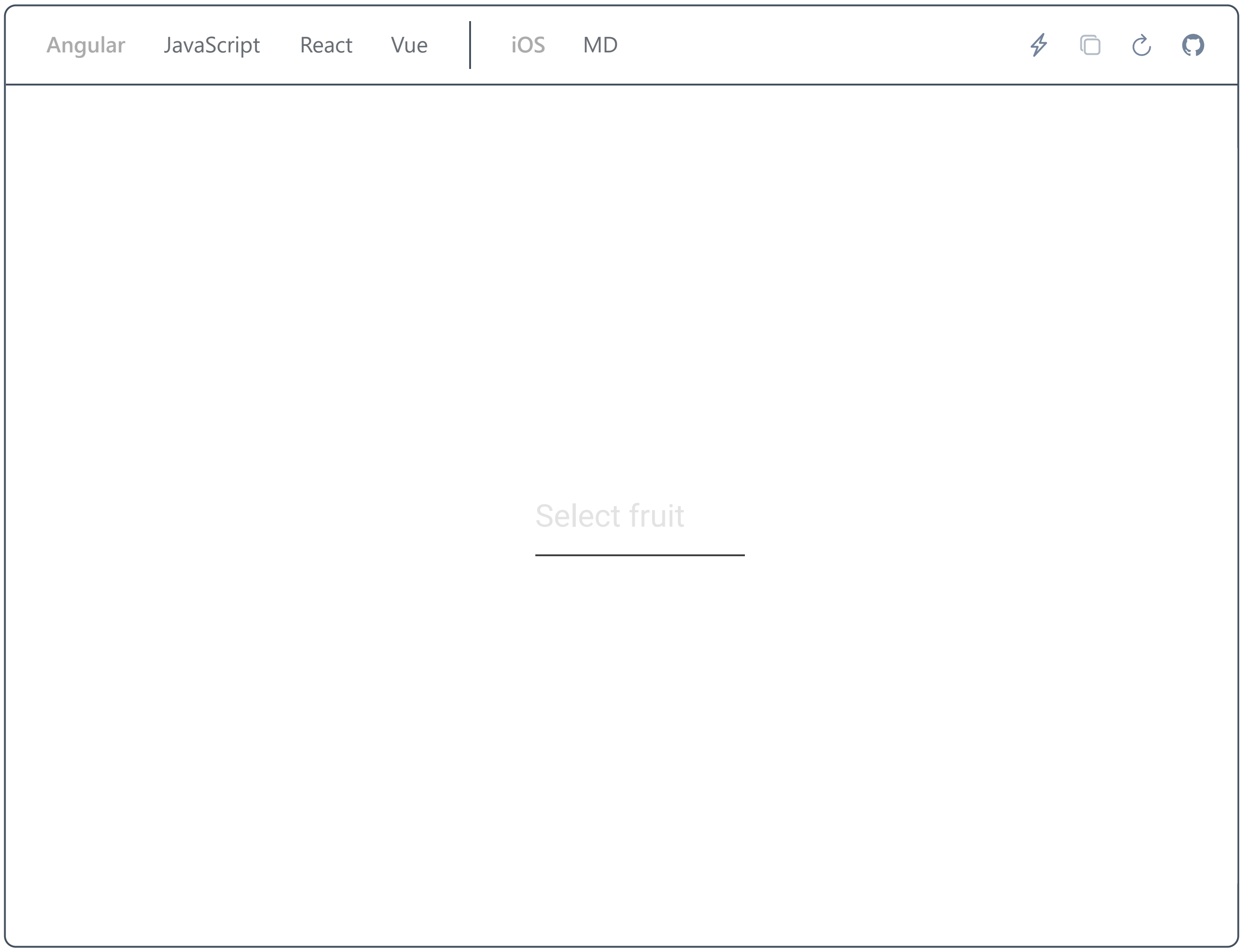


```
<ion-list>
  <ion-item>
    <ion-select placeholder="Select fruit">
      <ion-select-option value="apples">Apples</ion-select-option>
      <ion-select-option value="oranges">Oranges</ion-select-option>
      <ion-select-option value="bananas">Bananas</ion-select-option>
    </ion-select>
  </ion-item>
</ion-list>
```

Interfaces

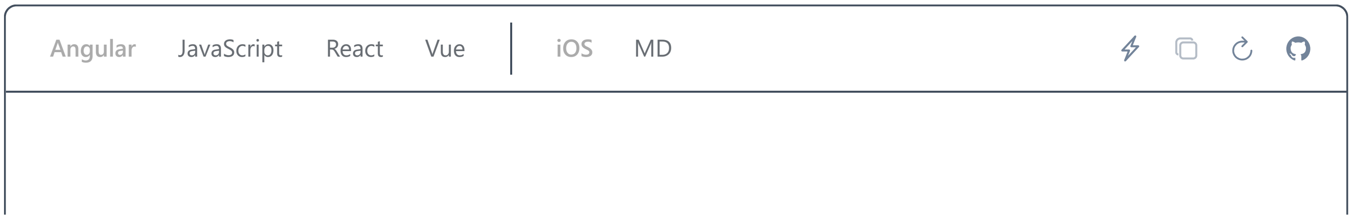
By default, select uses ion-alert to open up the overlay of options in an alert. The interface can be changed to use ion-action-sheet or ion-popover by passing `action-sheet` or `popover`, respectively, to the `interface` property. Read on to the other sections for the limitations of the different interfaces.

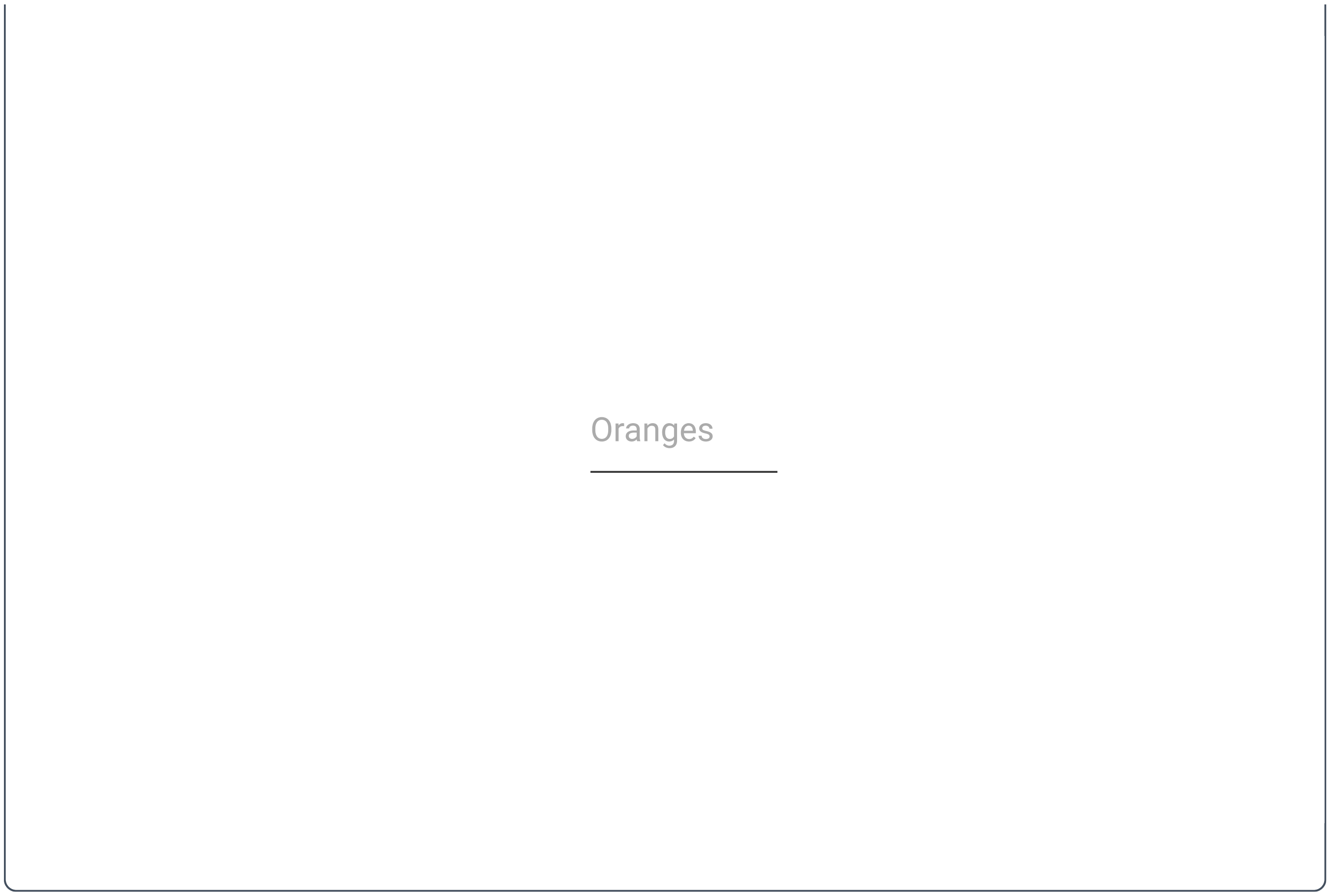
Action Sheet



```
<ion-list>
  <ion-item>
    <ion-select interface="action-sheet" placeholder="Select fruit">
      <ion-select-option value="apples">Apples</ion-select-option>
      <ion-select-option value="oranges">Oranges</ion-select-option>
      <ion-select-option value="bananas">Bananas</ion-select-option>
    </ion-select>
  </ion-item>
</ion-list>
```

Popover



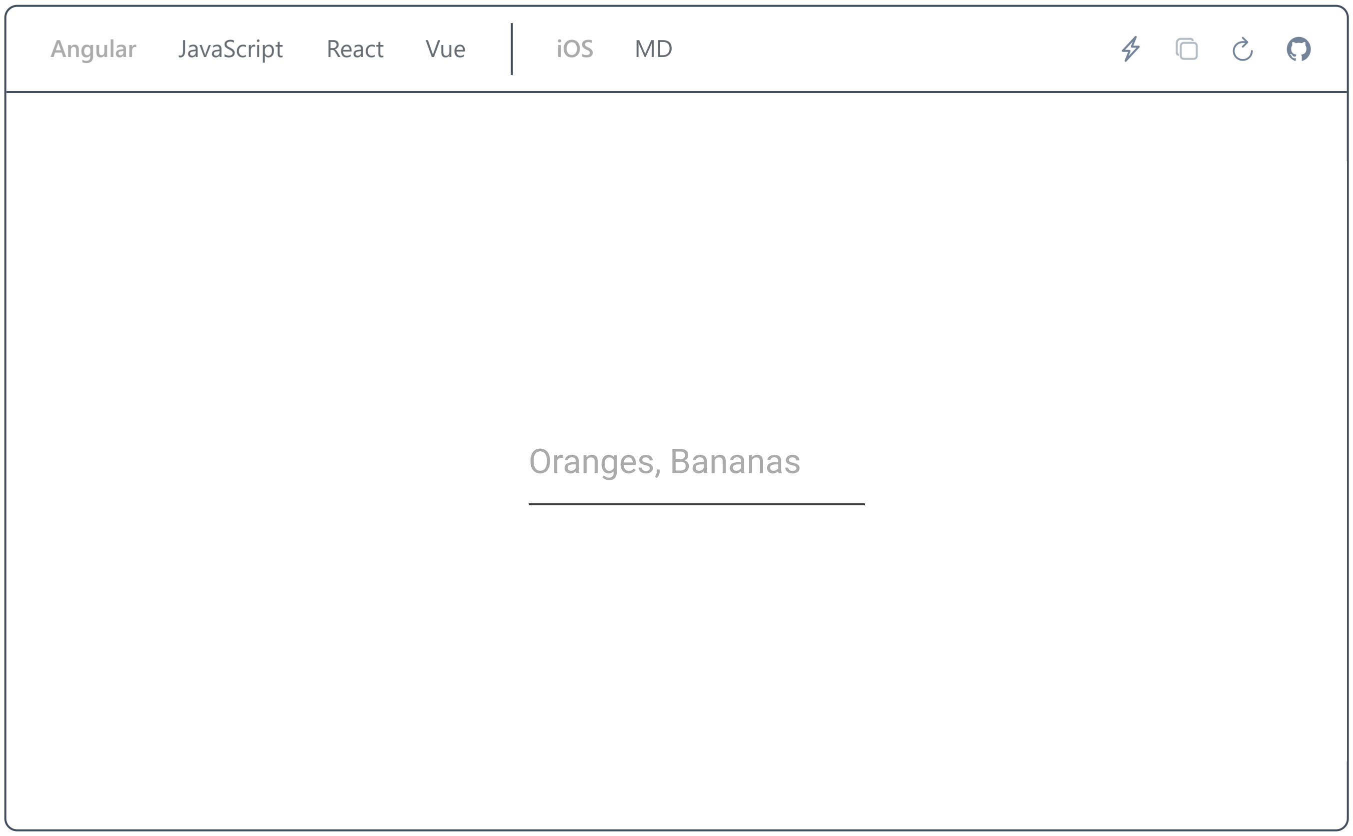


```
<ion-list>
  <ion-item>
    <ion-select interface="popover" placeholder="Select fruit">
      <ion-select-option value="apples">Apples</ion-select-option>
      <ion-select-option value="oranges">Oranges</ion-select-option>
      <ion-select-option value="bananas">Bananas</ion-select-option>
    </ion-select>
  </ion-item>
</ion-list>
```

Multiple Selection

By adding the `multiple` attribute to select, users are able to select multiple options. When multiple options can be selected, the alert overlay presents users with a checkbox styled list of options. The select component's value receives an array of all of the selected option values.

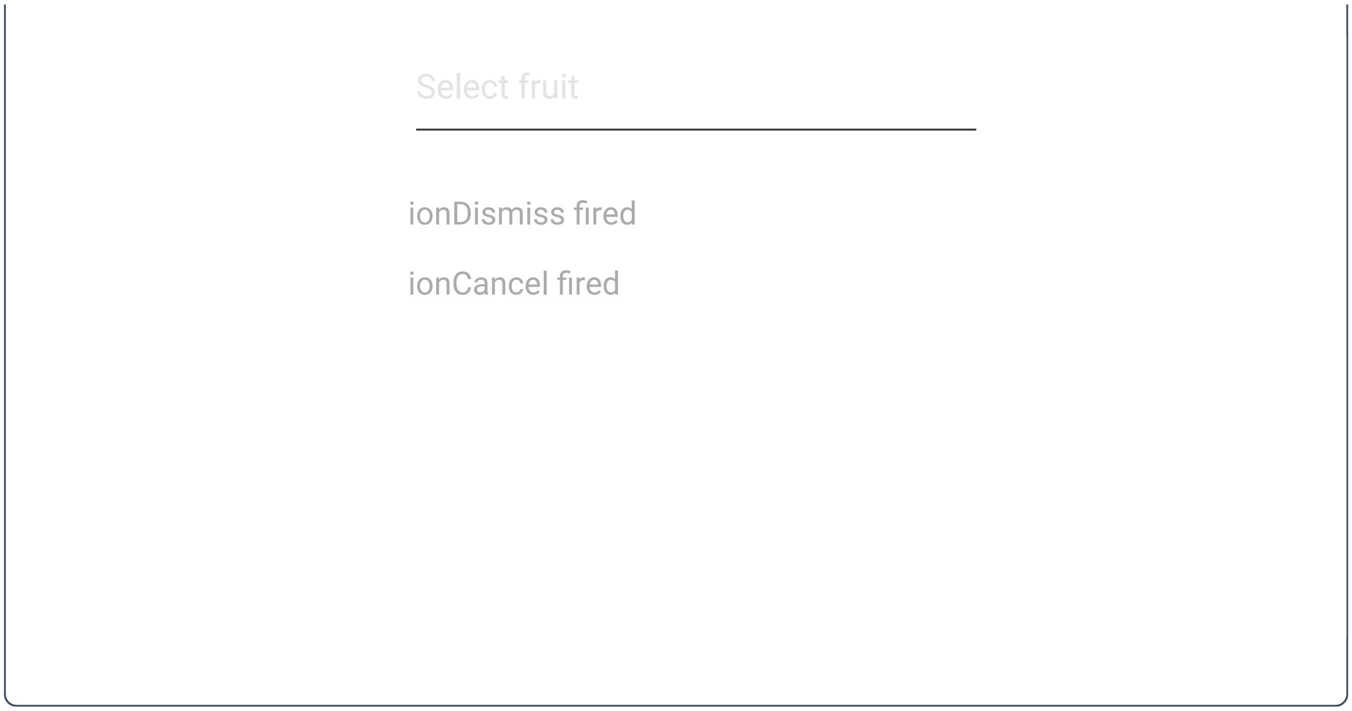
Note: the `action-sheet` and `popover` interfaces will not work with multiple selection.



```
<ion-list>
  <ion-item>
    <ion-select placeholder="Select all fruits that apply" [multiple]="true">
      <ion-select-option value="apples">Apples</ion-select-option>
      <ion-select-option value="oranges">Oranges</ion-select-option>
      <ion-select-option value="bananas">Bananas</ion-select-option>
    </ion-select>
  </ion-item>
</ion-list>
```

Responding to Interaction

The main ways of handling user interaction with the select are the `ionChange`, `ionDismiss`, and `ionCancel` events. See Events for more details on these and other events that select fires.



ionDismiss fired

ionCancel fired

src/app/example.component.html   TS src/app/example.component.ts

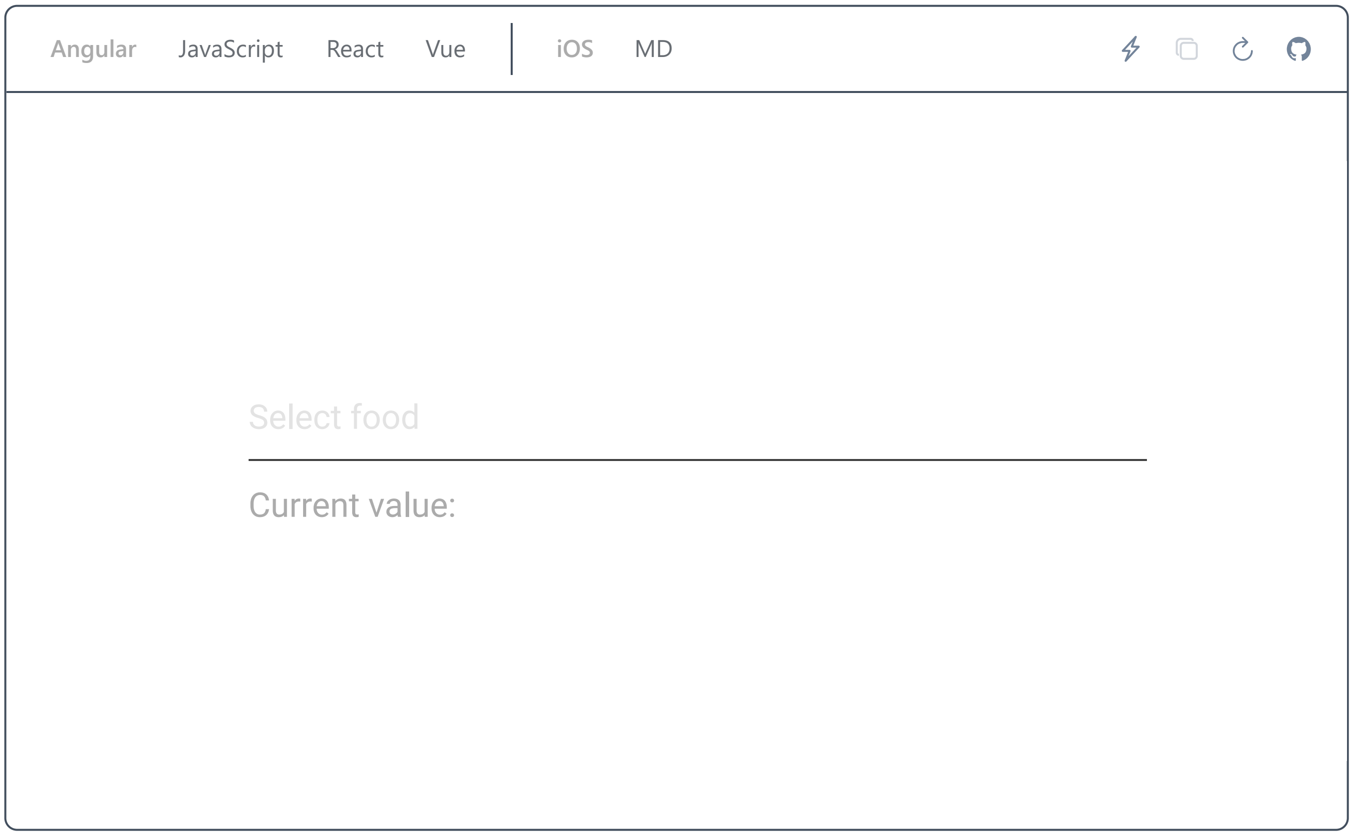
```
<ion-list>
  <ion-item>
    <ion-select
      placeholder="Select fruit"
      (ionChange)="handleChange($event)"
      (ionCancel)="pushLog('ionCancel fired')"
      (ionDismiss)="pushLog('ionDismiss fired')"
    >
      <ion-select-option value="apples">Apples</ion-select-option>
      <ion-select-option value="oranges">Oranges</ion-select-option>
      <ion-select-option value="bananas">Bananas</ion-select-option>
    </ion-select>
  </ion-item>
</ion-list>
<div class="ion-padding">
  <p *ngFor="let log of logs">{{ log }}</p>
</div>
```

Object Value References

When using objects for select values, it is possible for the identities of these objects to change if they are coming from a server or database, while the selected value's identity remains the same. For example, this can occur when an existing record with the desired object value is loaded into the select, but the newly retrieved select options now have different identities. This will result in the select appearing to have no value at all, even though the original selection in still intact.

By default, the select uses object equality (===) to determine if an option is selected. This can be overridden by providing a property name or a function to the `compareWith` property.

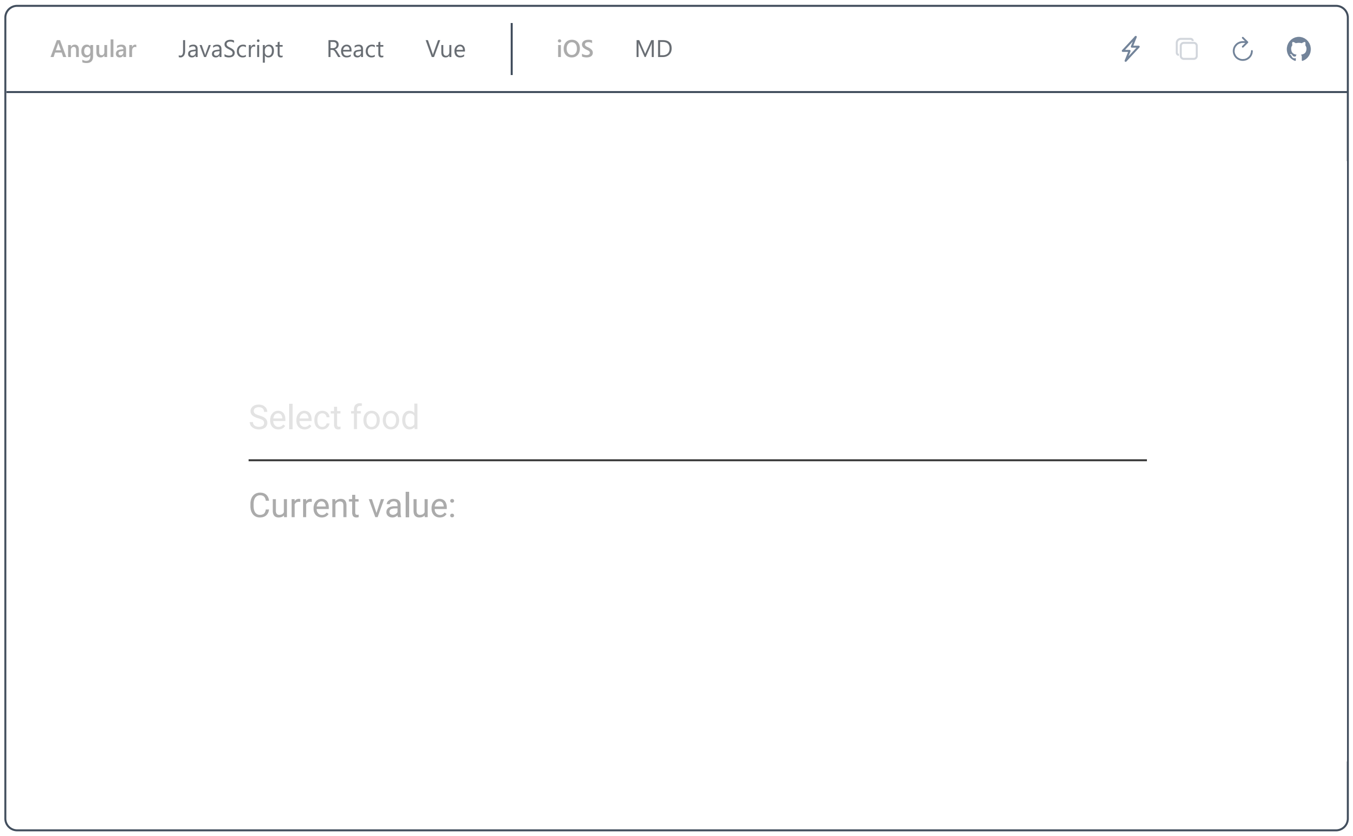
Using compareWith



src/app/example.component.html   TS src/app/example.component.ts

```
<ion-list>
  <ion-item>
    <ion-select
      [compareWith]="compareWith"
      placeholder="Select food"
      (ionChange)="handleChange($event)"
    >
      <ion-select-option *ngFor="let food of foods" [value]="food">{{ food.name }}
</ion-select-option>
    </ion-select>
  </ion-item>
  <ion-item lines="none">
    <ion-label>Current value: {{ currentFood | json }}</ion-label>
  </ion-item>
</ion-list>
```

Object Values and Multiple Selection



src/app/example.component.html TS src/app/example.component.ts

```
<ion-list>
  <ion-item>
    <ion-select
      [compareWith]="compareWith"
      placeholder="Select food"
      (ionChange)="handleChange($event)"
      [multiple]="true"
      >
        <ion-select-option *ngFor="let food of foods" [value]="food">{{ food.name }}
</ion-select-option>
      </ion-select>
    </ion-item>
    <ion-item lines="none">
      <ion-label>Current value: {{ currentFood | json }}</ion-label>
    </ion-item>
  </ion-list>
```

Select Buttons

The alert supports two buttons: `Cancel` and `OK`. Each button's text can be customized using the `cancelText` and `okText` properties.

The `action-sheet` and `popover` interfaces do not have an `OK` button, clicking on any of the options will automatically close the overlay and select that value. The `popover` interface does not have a `Cancel` button, clicking on the backdrop will close the overlay.

Angular JavaScript React Vue | iOS MD

```
<ion-list>
  <ion-item>
    <ion-label>Alert Interface</ion-label>
    <ion-select placeholder="Select fruit" okText="Choose Fruit" cancelText="Cancel
Choice">
      <ion-select-option value="apples">Apples</ion-select-option>
      <ion-select-option value="oranges">Oranges</ion-select-option>
      <ion-select-option value="bananas">Bananas</ion-select-option>
    </ion-select>
  </ion-item>
  <ion-item>
    <ion-label>Action Sheet Interface</ion-label>
    <ion-select interface="action-sheet" placeholder="Select fruit"
cancelText="Cancel Choice">
      <ion-select-option value="apples">Apples</ion-select-option>
      <ion-select-option value="oranges">Oranges</ion-select-option>
      <ion-select-option value="bananas">Bananas</ion-select-option>
    </ion-select>
    </ion-item>
  </ion-list>
```

Interface Options

Since select uses the alert, action sheet and popover interfaces, options can be passed to these components through the `interfaceOptions` property. This can be used to pass a custom header, subheader, css class, and more.

See the ion-alert docs, ion-action-sheet docs, and ion-popover docs for the properties that each interface accepts.

Note: `interfaceOptions` will not override `inputs` or `buttons` with the `alert` interface.

Angular JavaScript React Vue | iOS MD

src/app/example.component.html TS src/app/example.component.ts

```
  <ion-list>
    <ion-item>
      <ion-label>Alert</ion-label>
      <ion-select [interfaceOptions]="customAlertOptions" interface="alert"
  placeholder="Select One">
        <ion-select-option value="bacon">Bacon</ion-select-option>
        <ion-select-option value="onions">Onions</ion-select-option>
        <ion-select-option value="pepperoni">Pepperoni</ion-select-option>
      </ion-select>
    </ion-item>

    <ion-item>
      <ion-label>Popover</ion-label>
      <ion-select [interfaceOptions]="customPopoverOptions" interface="popover"
  placeholder="Select One">
        <ion-select-option value="brown">Brown</ion-select-option>
        <ion-select-option value="blonde">Blonde</ion-select-option>
        <ion-select-option value="red">Red</ion-select-option>
      </ion-select>
    </ion-item>
    <ion-item>
      <ion-label>Action Sheet</ion-label>
      <ion-select [interfaceOptions]="customActionSheetOptions" interface="action-
sheet" placeholder="Select One">
        <ion-select-option value="red">Red</ion-select-option>
        <ion-select-option value="green">Green</ion-select-option>
        <ion-select-option value="blue">Blue</ion-select-option>
      </ion-select>
    </ion-item>
  </ion-list>
```

Customization

There are two units that make up the Select component and each need to be styled separately. The `ion-select` element is represented on the view by the selected value(s), or placeholder if there is none, and dropdown icon. The interface, which is defined in the Interfaces section above, is the dialog that opens when clicking on the `ion-select`. The interface contains all of the options defined by adding `ion-select-option` elements. The following sections will go over the differences between styling these.

Styling Select Element

As mentioned, the `ion-select` element consists only of the value(s), or placeholder, and icon that is displayed on the view. To customize this, style using a combination of CSS and any of the CSS custom properties.

Alternatively, depending on the browser support needed, CSS shadow parts can be used to style the select. Notice that by using `::part`, any CSS property on the element can be targeted.

Angular JavaScript React Vue | iOS MD

src/app/example.component.html src/app/example.component.css src/app/example.component.ts

```
<ion-select placeholder="Select fruit">
  <ion-select-option value="apples">Apples</ion-select-option>
  <ion-select-option value="oranges">Oranges</ion-select-option>
  <ion-select-option value="bananas">Bananas</ion-select-option>
</ion-select>
```

Styling Select Interface

Customizing the interface dialog should be done by following the Customization section in that interface's documentation:

- Alert Customization
- Action Sheet Customization
- Popover Customization

However, the Select Option does set a class for easier styling and allows for the ability to pass a class to the overlay option, see the Select Options documentation for usage examples of customizing options.

Typeahead Component

Typeahead or autocomplete functionality can be built using existing Ionic components. We recommend using an `ion-modal` to make the best use of the available screen space.

Angular JavaScript React Vue | iOS MD

src/app/example.component.html TS src/app/example.component.ts src/app/typeahead.component.h

```
<ion-content color="light">
  <ion-list [inset]="true">
```

```
      <ion-item [button]="true" [detail]="false" id="select-fruits">
        <ion-label>Favorite Fruits</ion-label>
        <div slot="end" id="selected-fruits">{{ selectedFruitsText }}</div>
      </ion-item>
    </ion-list>
  </ion-content>

  <ion-modal trigger="select-fruits" #modal>
    <ng-template>
      <app-typeahead
        class="ion-page"
        title="Favorite Fruits"
        [items]="fruits"
        [selectedItems]="selectedFruits"
        (selectionChange)="fruitSelectionChanged($event)"
        (selectionCancel)="modal.dismiss()"
      ></app-typeahead>
    </ng-template>
  </ion-modal>
```

Interfaces

SelectChangeEventDetail

```
interface SelectChangeEventDetail<T = any> {
  value: T;
}
```

SelectCustomEvent

While not required, this interface can be used in place of the `CustomEvent` interface for stronger typing with Ionic events emitted from this component.

```
interface SelectCustomEvent<T = any> extends CustomEvent {
  detail: SelectChangeEventDetail<T>;
  target: HTMLIonSelectElement;
}
```

Properties

cancelText

Description	The text to display on the cancel button.
Attribute	`cancel-text`
Type	`string`
Default	`'Cancel'`

compareWith

Description	A property name or function used to compare object values
Attribute	`compare-with`
Type	`((currentValue: any, compareValue: any) => boolean) | null | string | undefined`
Default	`undefined`

disabled

Description	If `true`, the user cannot interact with the select.
Attribute	`disabled`
Type	`boolean`
Default	`false`

interface

Description	The interface the select should use: `action-sheet`, `popover` or `alert`.
Attribute	`interface`
Type	`"action-sheet" | "alert" | "popover"`
Default	`'alert'`

interfaceOptions

Description	Any additional options that the `alert`, `action-sheet` or `popover` interface can take. See the ion-alert docs, the ion-action-sheet docs and the ion-popover docs for the create options for each interface. Note: `interfaceOptions` will not override `inputs` or `buttons` with the `alert` interface.
Attribute	`interface-options`
Type	`any`
Default	`{}`

mode

Description	The mode determines which platform styles to use.
Attribute	`mode`
Type	`"ios" | "md"`
Default	`undefined`

multiple

Description	If `true` , the select can accept multiple values.
Attribute	`multiple`
Type	`boolean`
Default	`false`

name

Description	The name of the control, which is submitted with the form data.
Attribute	`name`
Type	`string`
Default	`this.inputId`

okText

Description	The text to display on the ok button.
Attribute	`ok-text`
Type	`string`
Default	`'OK'`

placeholder

Description	The text to display when the select is empty.
Attribute	`placeholder`
Type	`string | undefined`
Default	`undefined`

selectedText

Description	The text to display instead of the selected option's value.
Attribute	`selected-text`
Type	`null | string | undefined`
Default	`undefined`

value

Description	the value of the select.

Attribute	`value`
Type	`any`
Default	`undefined`

Events

Name	Description
`ionBlur`	Emitted when the select loses focus.
`ionCancel`	Emitted when the selection is cancelled.
`ionChange`	Emitted when the value has changed.
`ionDismiss`	Emitted when the overlay is dismissed.
`ionFocus`	Emitted when the select has focus.

Methods

open

Description	Open the select overlay. The overlay is either an alert, action sheet, or popover, depending on the `interface` property on the `ion-select`.
Signature	`open(event?: UIEvent) => Promise<any>`

CSS Shadow Parts

Name	Description
`icon`	The select icon container.
`placeholder`	The text displayed in the select when there is no value.
`text`	The displayed value of the select.

CSS Custom Properties

Name	Description
`--padding-bottom`	Bottom padding of the select
`--padding-end`	Right padding if direction is left-to-right, and left padding if direction is right-to-left of the select
`--padding-start`	Left padding if direction is left-to-right, and right padding if direction is right-to-left of the select
`--padding-top`	Top padding of the select
`--placeholder-color`	Color of the select placeholder text
`--placeholder-opacity`	Opacity of the select placeholder text

Slots

No slots available for this component.

Edit this page