Select

Select components are used for collecting user provided information from a list of options.

View as Markdown | Feedback | Bundle size | Source | WAI-ARIA | Material Design | Figma | Sketch

Basic select

Menus are positioned under their emitting elements, unless they are close to the bottom of the viewport.

Edit in Chat | Expand code

```
<FormControl fullWidth>
  <InputLabel id="demo-simple-select-label">Age</InputLabel>
  <Select
    labelId="demo-simple-select-label"
    id="demo-simple-select"
    value={age}
    label="Age"
    onChange={handleChange}
  >
    <MenuItem value={10}>Ten</MenuItem>
    <MenuItem value={20}>Twenty</MenuItem>
    <MenuItem value={30}>Thirty</MenuItem>
  </Select>
</FormControl>
```

Advanced features

The Select component is meant to be interchangeable with a native `<select>` element.

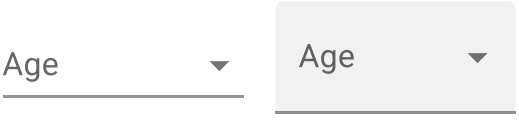
If you are looking for more advanced features, like combobox, multiselect, autocomplete, async or creatable support, head to the `Autocomplete` component. It's meant to be an improved version of the "react-select" and "downshift" packages.

Props

The Select component is implemented as a custom `<input>` element of the InputBase. It extends the text field components subcomponents, either the OutlinedInput, Input, or FilledInput, depending on the variant selected. It shares the same styles and many of the same props. Refer to the respective component's API page for details.

> ⚠ Unlike input components, the `placeholder` prop is not available in Select. To add a placeholder, refer to the placeholder section below.

Filled and standard variants

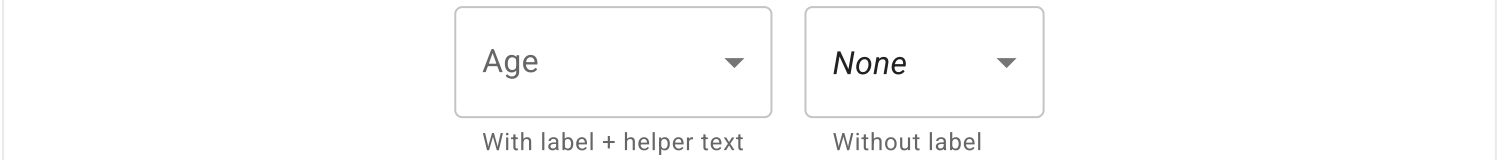


Edit in Chat JS TS Hide code

```
import * as React from 'react';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

export default function SelectVariants() {
  const [age, setAge] = React.useState('');

  const handleChange = (event: SelectChangeEvent) => {
    setAge(event.target.value);
  };

  return (
    <div>
      <FormControl variant="standard" sx={{ m: 1, minWidth: 120 }}>
        <InputLabel id="demo-simple-select-standard-label">Age</InputLabel>
        <Select
          labelId="demo-simple-select-standard-label"
          id="demo-simple-select-standard"
          value={age}
          onChange={handleChange}
          label="Age"
        >
```

```
        <MenuItem value="">
          <em>None</em>
        </MenuItem>
        <MenuItem value={10}>Ten</MenuItem>
        <MenuItem value={20}>Twenty</MenuItem>
        <MenuItem value={30}>Thirty</MenuItem>
      </Select>
    </FormControl>
    <FormControl variant="filled" sx={{ m: 1, minWidth: 120 }}>
      <InputLabel id="demo-simple-select-filled-label">Age</InputLabel>
      <Select
        labelId="demo-simple-select-filled-label"
        id="demo-simple-select-filled"
```

Labels and helper text



Edit in Chat JS TS Hide code

```
import * as React from 'react';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormHelperText from '@mui/material/FormHelperText';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

export default function SelectLabels() {
  const [age, setAge] = React.useState('');

  const handleChange = (event: SelectChangeEvent) => {
    setAge(event.target.value);
  };

  return (
    <div>
      <FormControl sx={{ m: 1, minWidth: 120 }}>
        <InputLabel id="demo-simple-select-helper-label">Age</InputLabel>
        <Select
          labelId="demo-simple-select-helper-label"
          id="demo-simple-select-helper"
          value={age}
          label="Age"
          onChange={handleChange}
        >
          <MenuItem value="">
            <em>None</em>
```

```
          </MenuItem>
          <MenuItem value={10}>Ten</MenuItem>
          <MenuItem value={20}>Twenty</MenuItem>
          <MenuItem value={30}>Thirty</MenuItem>
        </Select>
        <FormHelperText>With label + helper text</FormHelperText>
      </FormControl>
      <FormControl sx={{ m: 1, minWidth: 120 }}>
        <Select
          value={age}
```

> Note that when using FormControl with the outlined variant of the Select, you need to provide a label in two places: in the InputLabel component and in the `label` prop of the Select component (see the above demo).

Auto width

Edit in Chat | JS | TS | Hide code

```tsx
import * as React from 'react';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

export default function SelectAutoWidth() {
  const [age, setAge] = React.useState('');

  const handleChange = (event: SelectChangeEvent) => {
    setAge(event.target.value);
  };

  return (
    <div>
      <FormControl sx={{ m: 1, minWidth: 80 }}>
        <InputLabel id="demo-simple-select-autowidth-label">Age</InputLabel>
        <Select
          labelId="demo-simple-select-autowidth-label"
          id="demo-simple-select-autowidth"
          value={age}
          onChange={handleChange}
          autoWidth
          label="Age"
        >
```

```
          <MenuItem value="">
            <em>None</em>
          </MenuItem>
          <MenuItem value={20}>Twenty</MenuItem>
          <MenuItem value={21}>Twenty one</MenuItem>
          <MenuItem value={22}>Twenty one and a half</MenuItem>
        </Select>
      </FormControl>
    </div>
  );
}
```

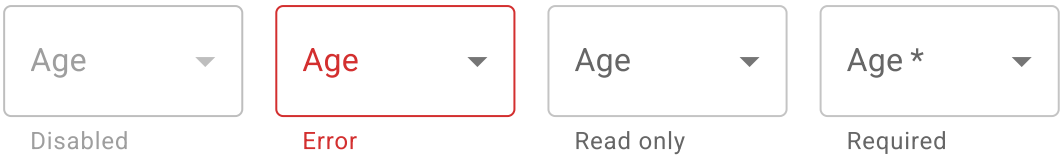
Small Size

Edit in Chat JS TS Hide code

```
import * as React from 'react';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

export default function SelectSmall() {
  const [age, setAge] = React.useState('');

  const handleChange = (event: SelectChangeEvent) => {
    setAge(event.target.value);
  };

  return (
    <FormControl sx={{ m: 1, minWidth: 120 }} size="small">
      <InputLabel id="demo-select-small-label">Age</InputLabel>
      <Select
        labelId="demo-select-small-label"
        id="demo-select-small"
        value={age}
        label="Age"
        onChange={handleChange}
      >
        <MenuItem value="">
          <em>None</em>
        </MenuItem>
        <MenuItem value={10}>Ten</MenuItem>
        <MenuItem value={20}>Twenty</MenuItem>
        <MenuItem value={30}>Thirty</MenuItem>
      </Select>
    </FormControl>
```

```
  );
}
```

Other props

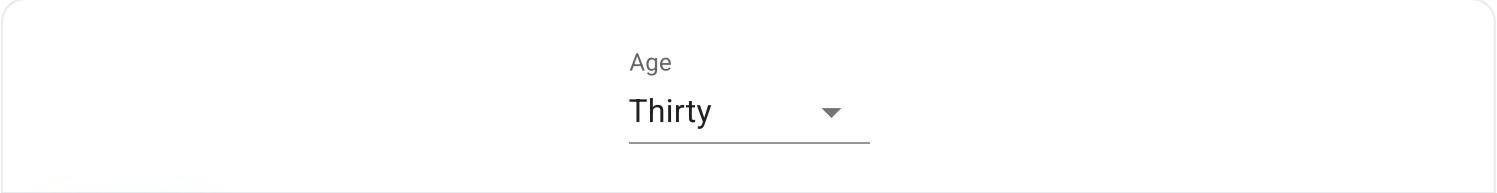


Edit in Chat | JS | TS | Hide code

```tsx
import * as React from 'react';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormHelperText from '@mui/material/FormHelperText';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

export default function SelectOtherProps() {
  const [age, setAge] = React.useState('');

  const handleChange = (event: SelectChangeEvent) => {
    setAge(event.target.value);
  };

  return (
    <div>
      <FormControl sx={{ m: 1, minWidth: 120 }} disabled>
        <InputLabel id="demo-simple-select-disabled-label">Age</InputLabel>
        <Select
          labelId="demo-simple-select-disabled-label"
          id="demo-simple-select-disabled"
          value={age}
          label="Age"
          onChange={handleChange}
        >
          <MenuItem value="">
            <em>None</em>
          </MenuItem>
          <MenuItem value={10}>Ten</MenuItem>
          <MenuItem value={20}>Twenty</MenuItem>
          <MenuItem value={30}>Thirty</MenuItem>
        </Select>
        <FormHelperText>Disabled</FormHelperText>
      </FormControl>
      <FormControl sx={{ m: 1, minWidth: 120 }} error>
        <InputLabel id="demo-simple-select-error-label">Age</InputLabel>
        <Select
          labelId="demo-simple-select-error-label"
```

Native select

As the user experience can be improved on mobile using the native select of the platform, we allow such pattern.



Edit in Chat JS TS Collapse code

```
import Box from '@mui/material/Box';
import InputLabel from '@mui/material/InputLabel';
import FormControl from '@mui/material/FormControl';
import NativeSelect from '@mui/material/NativeSelect';

export default function NativeSelectDemo() {
  return (
    <Box sx={{ minWidth: 120 }}>
      <FormControl fullWidth>
        <InputLabel variant="standard" htmlFor="uncontrolled-native">
          Age
        </InputLabel>
        <NativeSelect
          defaultValue={30}
          inputProps={{
            name: 'age',
            id: 'uncontrolled-native',
          }}
        >
          <option value={10}>Ten</option>
          <option value={20}>Twenty</option>
          <option value={30}>Thirty</option>
        </NativeSelect>
      </FormControl>
    </Box>
  );
}
```

TextField

The `TextField` wrapper component is a complete form control including a label, input and help text. You can find an example with the select mode in this section.

Customization

Here are some examples of customizing the component. You can learn more about this in the overrides documentation page.

The first step is to style the `InputBase` component. Once it's styled, you can either use it directly as a text field or provide it to the select `input` prop to have a `select` field. Notice that the `"standard"` variant is easier to customize, since it does not wrap the contents in a `fieldset`/`legend` markup.

Edit in Chat JS TS Hide code

```jsx
import * as React from 'react';
import { styled } from '@mui/material/styles';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select from '@mui/material/Select';
import NativeSelect from '@mui/material/NativeSelect';
import InputBase from '@mui/material/InputBase';

const BootstrapInput = styled(InputBase)(({ theme }) => ({
  'label + &': {
    marginTop: theme.spacing(3),
  },
  '& .MuiInputBase-input': {
    borderRadius: 4,
    position: 'relative',
    backgroundColor: (theme.vars ?? theme).palette.background.paper,
    border: '1px solid #ced4da',
    fontSize: 16,
    padding: '10px 26px 10px 12px',
    transition: theme.transitions.create(['border-color', 'box-shadow']),
    // Use the system font instead of the default Roboto font.
    fontFamily: [
      '-apple-system',
      'BlinkMacSystemFont',
      '"Segoe UI"',
      'Roboto',
      '"Helvetica Neue"',
      'Arial',
      'sans-serif',
      '"Apple Color Emoji"',
      '"Segoe UI Emoji"',
      '"Segoe UI Symbol"',
    ].join(','),
    '&:focus': {
      borderRadius: 4,
      borderColor: '#80bdff',
```

🎨 If you are looking for inspiration, you can check MUI Treasury's customization examples.

Multiple select

The `Select` component can handle multiple selections. It's enabled with the `multiple` prop.

Like with the single selection, you can pull out the new value by accessing `event.target.value` in the `onChange` callback. It's always an array.

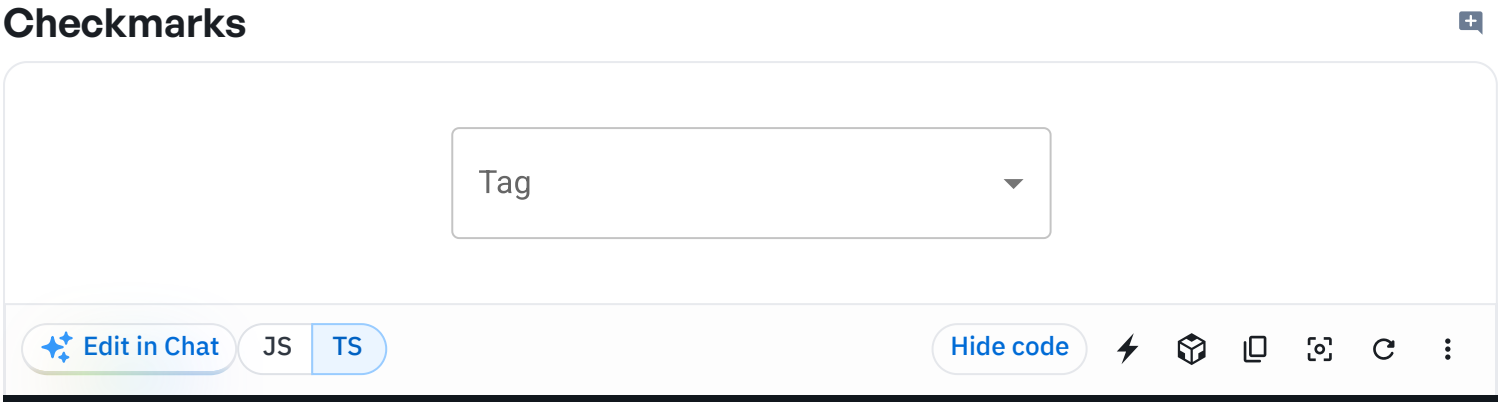
Default

Edit in Chat JS TS Hide code

```
import * as React from 'react';
import { Theme, useTheme } from '@mui/material/styles';
import OutlinedInput from '@mui/material/OutlinedInput';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

const ITEM_HEIGHT = 48;
const ITEM_PADDING_TOP = 8;
const MenuProps = {
  PaperProps: {
    style: {
      maxHeight: ITEM_HEIGHT * 4.5 + ITEM_PADDING_TOP,
      width: 250,
    },
  },
};

const names = [
  'Oliver Hansen',
  'Van Henry',
  'April Tucker',
  'Ralph Hubbard',
  'Omar Alexander',
  'Carlos Abbott',
  'Miriam Wagner',
  'Bradley Wilkerson',
  'Virginia Andrews',
  'Kelly Snyder',
```

```
];

function getStyles(name: string, personName: string[], theme: Theme) {
  return {
    fontWeight: personName.includes(name)
      ? theme.typography.fontWeightMedium
      : theme.typography.fontWeightRegular,
  };
```

Checkmarks



Edit in Chat JS TS Hide code

```
import * as React from 'react';
import OutlinedInput from '@mui/material/OutlinedInput';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import ListItemText from '@mui/material/ListItemText';
import Select, { SelectChangeEvent } from '@mui/material/Select';
import Checkbox from '@mui/material/Checkbox';

const ITEM_HEIGHT = 48;
const ITEM_PADDING_TOP = 8;
const MenuProps = {
  PaperProps: {
    style: {
      maxHeight: ITEM_HEIGHT * 4.5 + ITEM_PADDING_TOP,
      width: 250,
    },
  },
};

const names = [
  'Oliver Hansen',
  'Van Henry',
  'April Tucker',
  'Ralph Hubbard',
  'Omar Alexander',
  'Carlos Abbott',
  'Miriam Wagner',
  'Bradley Wilkerson',
  'Virginia Andrews',
  'Kelly Snyder',
];

export default function MultipleSelectCheckmarks() {
```

```tsx
  const [personName, setPersonName] = React.useState<string[]>([]);

  const handleChange = (event: SelectChangeEvent<typeof personName>) => {
    const {
```

Chip

Edit in Chat | JS | TS | Hide code

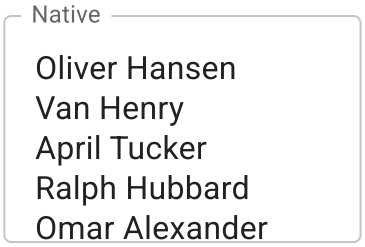
```tsx
import * as React from 'react';
import { Theme, useTheme } from '@mui/material/styles';
import Box from '@mui/material/Box';
import OutlinedInput from '@mui/material/OutlinedInput';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';
import Chip from '@mui/material/Chip';

const ITEM_HEIGHT = 48;
const ITEM_PADDING_TOP = 8;
const MenuProps = {
  PaperProps: {
    style: {
      maxHeight: ITEM_HEIGHT * 4.5 + ITEM_PADDING_TOP,
      width: 250,
    },
  },
};

const names = [
  'Oliver Hansen',
  'Van Henry',
  'April Tucker',
  'Ralph Hubbard',
  'Omar Alexander',
  'Carlos Abbott',
  'Miriam Wagner',
  'Bradley Wilkerson',
  'Virginia Andrews',
  'Kelly Snyder',
];

function getStyles(name: string, personName: readonly string[], theme: Theme) {
  return {
    fontWeight: personName.includes(name)
      ? theme.typography.fontWeightMedium
```

Placeholder

Edit in Chat JS TS Hide code

```tsx
import * as React from 'react';
import { Theme, useTheme } from '@mui/material/styles';
import OutlinedInput from '@mui/material/OutlinedInput';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

const ITEM_HEIGHT = 48;
const ITEM_PADDING_TOP = 8;
const MenuProps = {
  PaperProps: {
    style: {
      maxHeight: ITEM_HEIGHT * 4.5 + ITEM_PADDING_TOP,
      width: 250,
    },
  },
};

const names = [
  'Oliver Hansen',
  'Van Henry',
  'April Tucker',
  'Ralph Hubbard',
  'Omar Alexander',
  'Carlos Abbott',
  'Miriam Wagner',
  'Bradley Wilkerson',
  'Virginia Andrews',
  'Kelly Snyder',
];

function getStyles(name: string, personName: readonly string[], theme: Theme) {
  return {
    fontWeight: personName.includes(name)
      ? theme.typography.fontWeightMedium
      : theme.typography.fontWeightRegular,
  };
}
```

Native



Edit in Chat | JS | TS | Hide code

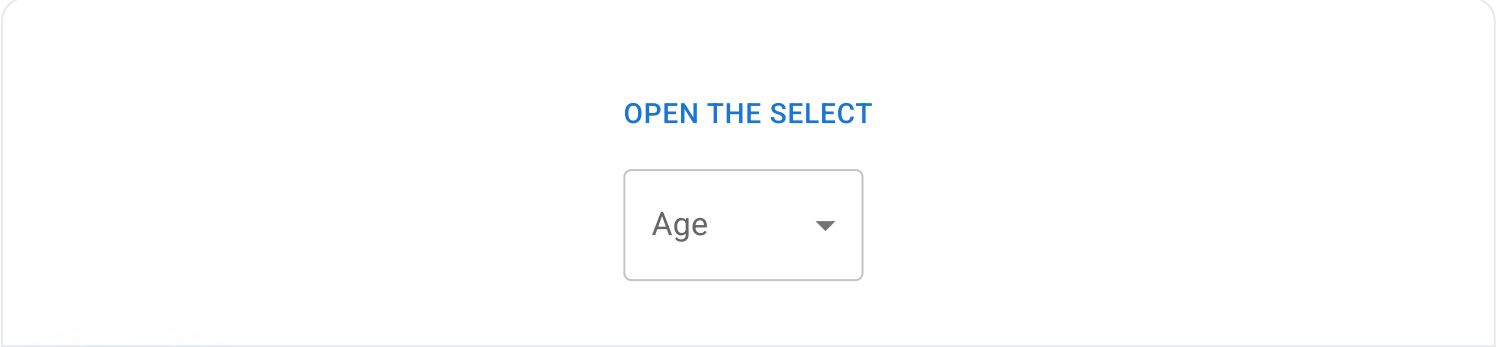
```tsx
import * as React from 'react';
import InputLabel from '@mui/material/InputLabel';
import FormControl from '@mui/material/FormControl';
import Select from '@mui/material/Select';

const names = [
  'Oliver Hansen',
  'Van Henry',
  'April Tucker',
  'Ralph Hubbard',
  'Omar Alexander',
  'Carlos Abbott',
  'Miriam Wagner',
  'Bradley Wilkerson',
  'Virginia Andrews',
  'Kelly Snyder',
];

export default function MultipleSelectNative() {
  const [personName, setPersonName] = React.useState<string[]>([]);
  const handleChangeMultiple = (event: React.ChangeEvent<HTMLSelectElement>) => {
    const { options } = event.target;
    const value: string[] = [];
    for (let i = 0, l = options.length; i < l; i += 1) {
      if (options[i].selected) {
        value.push(options[i].value);
      }
    }
    setPersonName(value);
  };

  return (
    <div>
      <FormControl sx={{ m: 1, minWidth: 120, maxWidth: 300 }}>
        <InputLabel shrink htmlFor="select-multiple-native">
          Native
        </InputLabel>
        <Select<string[]>
```

Controlling the open state

You can control the open state of the select with the `open` prop. Alternatively, it is also possible to set the initial (uncontrolled) open state of the component with the `defaultOpen` prop.

- A component is **controlled** when it's managed by its parent using props.
- A component is **uncontrolled** when it's managed by its own local state.

Learn more about controlled and uncontrolled components in the React documentation.



Edit in Chat | JS | TS | Hide code

```tsx
import * as React from 'react';
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';
import Button from '@mui/material/Button';

export default function ControlledOpenSelect() {
  const [age, setAge] = React.useState<string | number>('');
  const [open, setOpen] = React.useState(false);

  const handleChange = (event: SelectChangeEvent<typeof age>) => {
    setAge(event.target.value);
  };

  const handleClose = () => {
    setOpen(false);
  };

  const handleOpen = () => {
    setOpen(true);
  };

  return (
    <div>
      <Button sx={{ display: 'block', mt: 2 }} onClick={handleOpen}>
        Open the select
      </Button>
      <FormControl sx={{ m: 1, minWidth: 120 }}>
        <InputLabel id="demo-controlled-open-select-label">Age</InputLabel>
        <Select
          labelId="demo-controlled-open-select-label"
```

```
          id="demo-controlled-open-select"
          open={open}
          onClose={handleClose}
          onOpen={handleOpen}
          value={age}
          label="Age"
```

With a dialog

While it's discouraged by the Material Design guidelines, you can use a select inside a dialog.

OPEN SELECT DIALOG

Edit in Chat | JS | TS | Hide code

```tsx
import * as React from 'react';
import Box from '@mui/material/Box';
import Button from '@mui/material/Button';
import Dialog from '@mui/material/Dialog';
import DialogActions from '@mui/material/DialogActions';
import DialogContent from '@mui/material/DialogContent';
import DialogTitle from '@mui/material/DialogTitle';
import InputLabel from '@mui/material/InputLabel';
import OutlinedInput from '@mui/material/OutlinedInput';
import MenuItem from '@mui/material/MenuItem';
import FormControl from '@mui/material/FormControl';
import Select, { SelectChangeEvent } from '@mui/material/Select';

export default function DialogSelect() {
  const [open, setOpen] = React.useState(false);
  const [age, setAge] = React.useState<number | string>('');

  const handleChange = (event: SelectChangeEvent<typeof age>) => {
    setAge(Number(event.target.value) || '');
  };

  const handleClickOpen = () => {
    setOpen(true);
  };

  const handleClose = (event: React.SyntheticEvent<unknown>, reason?: string) => {
    if (reason !== 'backdropClick') {
      setOpen(false);
    }
  };

  return (
    <div>
      <Button onClick={handleClickOpen}>Open select dialog</Button>
      <Dialog disableEscapeKeyDown open={open} onClose={handleClose}>
```

```
        <DialogTitle>Fill the form</DialogTitle>
        <DialogContent>
          <Box component="form" sx={{ display: 'flex', flexWrap: 'wrap' }}>
```

Grouping

Display categories with the `ListSubheader` component or the native `<optgroup>` element.

Edit in Chat | JS | TS | Hide code

```
import InputLabel from '@mui/material/InputLabel';
import MenuItem from '@mui/material/MenuItem';
import ListSubheader from '@mui/material/ListSubheader';
import FormControl from '@mui/material/FormControl';
import Select from '@mui/material/Select';

export default function GroupedSelect() {
  return (
    <div>
      <FormControl sx={{ m: 1, minWidth: 120 }}>
        <InputLabel htmlFor="grouped-native-select">Grouping</InputLabel>
        <Select native defaultValue="" id="grouped-native-select" label="Grouping">
          <option aria-label="None" value="" />
          <optgroup label="Category 1">
            <option value={1}>Option 1</option>
            <option value={2}>Option 2</option>
          </optgroup>
          <optgroup label="Category 2">
            <option value={3}>Option 3</option>
            <option value={4}>Option 4</option>
          </optgroup>
        </Select>
      </FormControl>
      <FormControl sx={{ m: 1, minWidth: 120 }}>
        <InputLabel id="grouped-select-label" htmlFor="grouped-select">
          Grouping
        </InputLabel>
        <Select
          defaultValue=""
          id="grouped-select"
          label="Grouping"
          SelectDisplayProps={{
            'aria-labelledby': 'grouped-select-label',
          }}
        >
          <MenuItem value="">
```

```
      <em>None</em>
    </MenuItem>
```

⚠ If you wish to wrap the ListSubheader in a custom component, you'll have to annotate it so Material UI can handle it properly when determining focusable elements.

You have two options for solving this: Option 1: Define a static boolean field called `muiSkipListHighlight` on your component function, and set it to `true`:

Copy

```
function MyListSubheader(props: ListSubheaderProps) {
  return <ListSubheader {...props} />;
}

MyListSubheader.muiSkipListHighlight = true;
export default MyListSubheader;

// elsewhere:

return (
  <Select>
    <MyListSubheader>Group 1</MyListSubheader>
    <MenuItem value={1}>Option 1</MenuItem>
    <MenuItem value={2}>Option 2</MenuItem>
    <MyListSubheader>Group 2</MyListSubheader>
    <MenuItem value={3}>Option 3</MenuItem>
    <MenuItem value={4}>Option 4</MenuItem>
    {/* ... */}
  </Select>
```

Option 2: Place a `muiSkipListHighlight` prop on each instance of your component. The prop doesn't have to be forwarded to the ListSubheader, nor present in the underlying DOM element. It just has to be placed on a component that's used as a subheader.

Copy

```
export default function MyListSubheader(
  props: ListSubheaderProps & { muiSkipListHighlight: boolean },
) {
  const { muiSkipListHighlight, ...other } = props;
  return <ListSubheader {...other} />;
}

// elsewhere:

return (
  <Select>
    <MyListSubheader muiSkipListHighlight>Group 1</MyListSubheader>
    <MenuItem value={1}>Option 1</MenuItem>
    <MenuItem value={2}>Option 2</MenuItem>
    <MyListSubheader muiSkipListHighlight>Group 2</MyListSubheader>
    <MenuItem value={3}>Option 3</MenuItem>
    <MenuItem value={4}>Option 4</MenuItem>
    {/* ... */}
```

```
</Select>
```

We recommend the first option as it doesn't require updating all the usage sites of the component.

Keep in mind this is **only necessary** if you wrap the ListSubheader in a custom component. If you use the ListSubheader directly, **no additional code is required**.

Accessibility

To properly label your `Select` input you need an extra element with an `id` that contains a label. That `id` needs to match the `labelId` of the `Select`, for example:

```
<InputLabel id="label">Age</InputLabel>
<Select labelId="label" id="select" value="20">
  <MenuItem value="10">Ten</MenuItem>
  <MenuItem value="20">Twenty</MenuItem>
</Select>
```

Copy

Alternatively a `TextField` with an `id` and `label` creates the proper markup and ids for you:

```
<TextField id="select" label="Age" value="20" select>
  <MenuItem value="10">Ten</MenuItem>
  <MenuItem value="20">Twenty</MenuItem>
</TextField>
```

Copy

For a native select, you should mention a label by giving the value of the `id` attribute of the select element to the `InputLabel`'s `htmlFor` attribute:

```
<InputLabel htmlFor="select">Age</InputLabel>
<NativeSelect id="select">
  <option value="10">Ten</option>
  <option value="20">Twenty</option>
</NativeSelect>
```

Copy

API

See the documentation below for a complete reference to all of the props and classes available to the components mentioned here.

- `<NativeSelect />`
- `<Select />`

‹ Rating

Slider ›