# Radio

Radio.

## When To Use

- Used to select a single state from multiple options.
- The difference from Select is that Radio is visible to the user and can facilitate the comparison of choice, which means there shouldn't be too many of them.

## Examples

Radio

## Basic

The simplest use.

```
import React from 'react';
import { Radio } from 'antd';

const App: React.FC = () => <Radio>Radio</Radi

export default App;
```

A B C D

## Radio Group

A group of radio components.

```
import React, { useState } from 'react';
import type { RadioChangeEvent } from 'antd';
import { Radio } from 'antd';

const App: React.FC = () => {
  const [value, setValue] = useState(1);

  const onChange = (e: RadioChangeEvent) => {
    console.log('radio checked', e.target.valu
    setValue(e.target.value);
  };

  return (
    <Radio.Group onChange={onChange} value={va
      <Radio value={1}>A</Radio>
      <Radio value={2}>B</Radio>
      <Radio value={3}>C</Radio>
      <Radio value={4}>D</Radio>
    </Radio.Group>
  );
};

export default App;
```

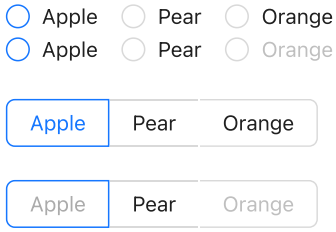
Disabled Disabled

Toggle disabled

## disabled

Radio unavailable.

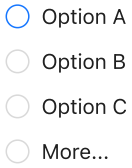
```
import React, { useState } from 'react';
import { Button, Radio } from 'antd';

const App: React.FC = () => {
  const [disabled, setDisabled] = useState(tru

  const toggleDisabled = () => {
    setDisabled(!disabled);
  };

  return (
    <>
      <Radio defaultChecked={false} disabled={
        Disabled
      </Radio>
      <Radio defaultChecked disabled={disablec
        Disabled
      </Radio>
      <br />
      <Button type="primary" onClick={toggleDi
        Toggle disabled
      </Button>
    </>
  );
};

export default App;
```




### Radio.Group group - optional

Render radios by configuring `options`. Radio type can also be set through the `optionType` parameter.

```
import React, { useState } from 'react';
import type { RadioChangeEvent } from 'antd';
import { Radio } from 'antd';

const plainOptions = ['Apple', 'Pear', 'Orange
const options = [
  { label: 'Apple', value: 'Apple' },
  { label: 'Pear', value: 'Pear' },
  { label: 'Orange', value: 'Orange' },
];
const optionsWithDisabled = [
  { label: 'Apple', value: 'Apple' },
  { label: 'Pear', value: 'Pear' },
  { label: 'Orange', value: 'Orange', disablec
];

const App: React.FC = () => {
  const [value1, setValue1] = useState('Apple'
  const [value2, setValue2] = useState('Apple'
  const [value3, setValue3] = useState('Apple'
  const [value4, setValue4] = useState('Apple'

  const onChange1 = ({ target: { value } }: Rc
    console.log('radio1 checked', value);
    setValue1(value);
  };

  const onChange2 = ({ target: { value } }: Rc
    console.log('radio2 checked', value);
    setValue2(value);
  };

  const onChange3 = ({ target: { value } }: Rc
    console.log('radio3 checked', value);
    setValue3(value);
  };

  const onChange4 = ({ target: { value } }: Rc
    console.log('radio4 checked', value);
    setValue4(value);
  };

  return (
    <>
      <Radio.Group options={plainOptions} onCh
      <br />
      <Radio.Group options={optionsWithDisable
      <br />
      <br />
      <Radio.Group options={options} onChange=
      <br />
      <br />
      <Radio.Group
        options={optionsWithDisabled}
        onChange={onChange4}
```




### Vertical Radio.Group

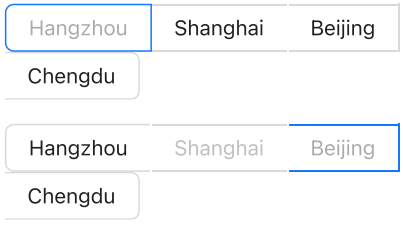
Vertical Radio.Group, with more radios.

```
import React, { useState } from 'react';
import type { RadioChangeEvent } from 'antd';
import { Input, Radio, Space } from 'antd';

const App: React.FC = () => {
  const [value, setValue] = useState(1);

  const onChange = (e: RadioChangeEvent) => {
    console.log('radio checked', e.target.valu
    setValue(e.target.value);
  };

  return (
    <Radio.Group onChange={onChange} value={vc
      <Space direction="vertical">
        <Radio value={1}>Option A</Radio>
        <Radio value={2}>Option B</Radio>
        <Radio value={3}>Option C</Radio>
        <Radio value={4}>
          More...
          {value === 4 ? <Input style={{ widtl
        </Radio>
      </Space>
    </Radio.Group>
  );
};

export default App;
```


## Radio.Group with name

Passing the `name` property to all `input[type="radio"]` that are in the same Radio.Group. It is usually used to let the browser see your Radio.Group as a real "group" and keep the default behavior. For example, using left/right keyboard arrow to change your selection that in the same Radio.Group.

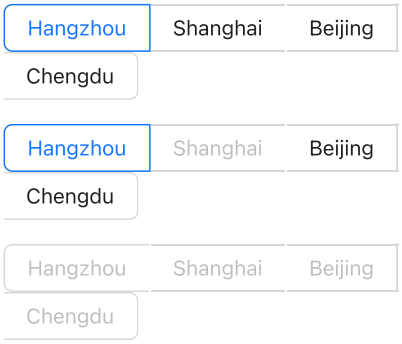
```
import React from 'react';
import { Radio } from 'antd';

const App: React.FC = () => (
  <Radio.Group name="radiogroup" defaultValue=
    <Radio value={1}>A</Radio>
    <Radio value={2}>B</Radio>
    <Radio value={3}>C</Radio>
    <Radio value={4}>D</Radio>
  </Radio.Group>
);

export default App;
```




## Solid radio button

Solid radio button style.

```
import React from 'react';
import { Radio } from 'antd';

const App: React.FC = () => (
  <>
    <Radio.Group defaultValue="a" buttonStyle=
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b">Shanghai</Radio.
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
    <Radio.Group defaultValue="c" buttonStyle=
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b" disabled>
        Shanghai
      </Radio.Button>
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
  </>
);

export default App;
```




## radio style

The combination of radio button style.

```
import React from 'react';
import type { RadioChangeEvent } from 'antd';
import { Radio } from 'antd';

const onChange = (e: RadioChangeEvent) => {
  console.log(`radio checked:${e.target.value}
};

const App: React.FC = () => (
  <>
    <Radio.Group onChange={onChange} defaultVa
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b">Shanghai</Radio.
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
    <Radio.Group onChange={onChange} defaultVa
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b" disabled>
        Shanghai
      </Radio.Button>
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
    <Radio.Group disabled onChange={onChange}
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b">Shanghai</Radio.
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
  </>
);

export default App;
```


### Size

There are three sizes available: large, medium, and small. It can coordinate with input box.

```
import React from 'react';
import { Radio } from 'antd';

const App: React.FC = () => (
  <>
    <Radio.Group defaultValue="a" size="large'
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b">Shanghai</Radio.
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
    <Radio.Group defaultValue="a" style={{ mar
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b">Shanghai</Radio.
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
    <Radio.Group defaultValue="a" size="small'
      <Radio.Button value="a">Hangzhou</Radio.
      <Radio.Button value="b">Shanghai</Radio.
      <Radio.Button value="c">Beijing</Radio.E
      <Radio.Button value="d">Chengdu</Radio.E
    </Radio.Group>
  </>
);

export default App;
```

## API

### Radio/Radio.Button

| Property | Description | Type | Default |
| --- | --- | --- | --- |
| autoFocus | Whether get focus when component mounted | boolean | false |
| checked | Specifies whether the radio is selected | boolean | false |
| defaultChecked | Specifies the initial state: whether or not the radio is selected | boolean | false |
| disabled | Disable radio | boolean | false |

| Property | Description | Type | Default |
| --- | --- | --- | --- |
| value | According to value for comparison, to determine whether the selected | any | - |

### RadioGroup

Radio group can wrap a group of `Radio` 。

| Property | Description | Type | Default | Version |
| --- | --- | --- | --- | --- |
| buttonStyle | The style type of radio button | `outline` \| `solid` | `outline` | |
| defaultValue | Default selected value | any | - | |
| disabled | Disable all radio buttons | boolean | false | |
| name | The `name` property of all `input[type="radio"]` children | string | - | |
| options | Set children optional | string[] \| number[] \| Array<{ label: ReactNode; value: string; disabled?: boolean; }> | - | |
| optionType | Set Radio optionType | `default` \| `button` | `default` | 4.4.0 |
| size | The size of radio button style | `large` \| `middle` \| `small` | - | |
| value | Used for setting the currently selected value | any | - | |
| onChange | The callback function that is triggered when the state changes | function(e:Event) | - | |

## Methods

### Radio

| Name | Description |
| --- | --- |
| blur() | Remove focus |
| focus() | Get focus |

## Design Token

▼ Global Token

| Token Name | Description | Type | Default Value |
| --- | --- | --- | --- |
| colorBgContainer | Container background color, e.g: default button, input box, etc. Be sure not to confuse this with `colorBgElevated`. | `string` | #ffffff |
| colorBgContainerDisabled | Control the background color of container in disabled state. | `string` | rgba(0, 0, 0, 0.04) |
| colorBorder | Default border color, used to separate different elements, such as: form separator, card separator, etc. | `string` | #d9d9d9 |
| colorPrimary | Brand color is one of the most direct visual elements to reflect the characteristics and communication of the product. After you have selected the brand color, we will automatically generate a complete color palette and assign it effective design semantics. | `string` | #1677ff |
| colorPrimaryActive | Dark active state under the main color gradient. | `string` | #0958d9 |
| colorPrimaryBorder | The stroke color under the main color gradient, used on the stroke of components such as Slider. | `string` | #91caff |
| colorPrimaryHover | Hover state under the main color gradient. | `string` | #4096ff |
| colorText | Default text color which comply with W3C standards, and this color is also the darkest neutral color. | `string` | rgba(0, 0, 0, 0.88) |
| colorTextDisabled | Control the color of text in disabled state. | `string` | rgba(0, 0, 0, 0.25) |
| colorTextLightSolid | Control the highlight color of text with background color, such as the text in Primary Button components. | `string` | #fff |
| colorWhite | Pure white color don't changed by theme | `string` | #fff |
| borderRadius | Border radius of base components | `number` | 6 |
| borderRadiusLG | LG size border radius, used in some large border radius components, such as Card, Modal and other components. | `number` | 8 |
| borderRadiusSM | SM size border radius, used in small size components, such as Button, Input, Select and other input components in small size | `number` | 4 |
| controlHeight | The height of the basic controls such as buttons and input boxes in Ant Design | `number` | 32 |

| Token Name | Description | Type | Default Value |
| --- | --- | --- | --- |
| controlHeightLG | LG component height | number | 40 |
| controlHeightSM | SM component height | number | 24 |
| controlItemBgActiveDisabled | Control the background color of control component item when active and disabled. | string | rgba(0, 0, 0, 0.15) |
| controlOutline | Control the outline color of input component. | string | rgba(5, 145, 255, 0.1) |
| controlOutlineWidth | Control the outline width of input component. | number | 2 |
| fontFamily | The font family of Ant Design prioritizes the default interface font of the system, and provides a set of alternative font libraries that are suitable for screen display to maintain the readability and readability of the font under different platforms and browsers, reflecting the friendly, stable and professional characteristics. | string | -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, 'Helvetica Neue', Arial, 'Noto Sans', sans-serif, 'Apple Color Emoji', 'Segoe UI Emoji', 'Segoe UI Symbol', 'Noto Color Emoji' |
| fontSize | The most widely used font size in the design system, from which the text gradient will be derived. | number | 14 |
| fontSizeLG | Large font size | number | 16 |
| lineHeight | Line height of text. | number | 1.5714285714285714 |
| lineType | Border style of base components | string | solid |
| lineWidth | Border width of base components | number | 1 |
| lineWidthFocus | Control the width of the line when the component is in focus state. | number | 4 |
| marginXS | Control the margin of an element, with a small size. | number | 8 |
| motionDurationMid | Motion speed, medium speed. Used for medium element animation interaction. | string | 0.2s |
| motionDurationSlow | Motion speed, slow speed. Used for large element animation interaction. | string | 0.3s |
| motionEaseInOut | Preset motion curve. | string | cubic-bezier(0.645, 0.045, 0.355, 1) |
| motionEaseInOutCirc | Preset motion curve. | string | cubic-bezier(0.78, 0.14, 0.15, 0.86) |
| padding | Control the padding of the element. | number | 16 |
| paddingXS | Control the extra small padding of the | number | 8 |

| Token Name | Description | Type | Default Value |
| --- | --- | --- | --- |
| | element. | | |
| wireframe | Used to change the visual effect of the component to wireframe, if you need to use the V4 effect, you need to enable the configuration item | `boolean` | false |