InputNumber

Enter a number within certain range with the mouse or keyboard.

When To Use

When a numeric value needs to be provided.

Examples

Basic

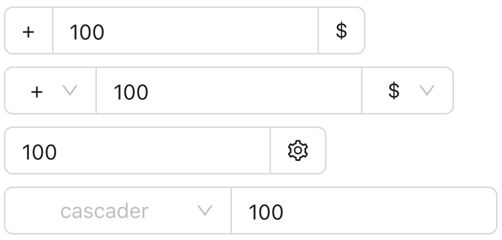
Numeric-only input box.

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

const onChange = (value: number) => {
  console.log('changed', value);
};

const App: React.FC = () => <InputNumber min={

export default App;
```



Pre / Post tab

Using pre & post tabs example.

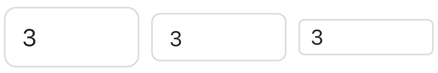
```
import React from 'react';
import { SettingOutlined } from '@ant-design/i
import { Cascader, InputNumber, Select, Space

const { Option } = Select;

const selectBefore = (
  <Select defaultValue="add" style={{ width: 6
    <Option value="add">+</Option>
    <Option value="minus">-</Option>
  </Select>
);
const selectAfter = (
  <Select defaultValue="USD" style={{ width: 6
    <Option value="USD">$</Option>
    <Option value="EUR">€</Option>
    <Option value="GBP">£</Option>
    <Option value="CNY">¥</Option>
  </Select>
);

const App: React.FC = () => (
  <Space direction="vertical">
    <InputNumber addonBefore="+" addonAfter="$
    <InputNumber addonBefore={selectBefore} ac
    <InputNumber addonAfter={<SettingOutlined
    <InputNumber
      addonBefore={<Cascader placeholder="casc
      defaultValue={100}
    />
  </Space>
);

export default App;
```



Sizes

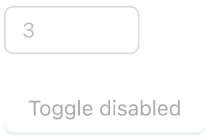
There are three sizes available to a numeric input box. By default, the size is `32px`. The two additional sizes are `large` and `small` which means `40px` and `24px`, respectively.

```
import React from 'react';
import { InputNumber, Space } from 'antd';

const onChange = (value: number) => {
  console.log('changed', value);
};

const App: React.FC = () => (
  <Space>
    <InputNumber size="large" min={1} max={10
    <InputNumber min={1} max={100000} defaultV
    <InputNumber size="small" min={1} max={10
  </Space>
);

export default App;
```



Disabled

Click the button to toggle between available and disabled states.

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

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

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

  return (
    <>
      <InputNumber min={1} max={10} disabled={
      <div style={{ marginTop: 20 }}>
        <Button onClick={toggle} type="primary
          Toggle disabled
        </Button>
      </div>
    </>
  );
};

export default App;
```

1.00000000000000

High precision decimals

Use `stringMode` to support high precision decimals support. `onChange` will return string value instead. You need polyfill of BigInt if browser not support.

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

const onChange = (value: string) => {
  console.log('changed', value);
};

const App: React.FC = () => (
  <InputNumber<string>
    style={{ width: 200 }}
    defaultValue="1"
    min="0"
    max="10"
    step="0.00000000000001"
    onChange={onChange}
    stringMode
  />
);

export default App;
```

3 Toggle keyboard

Keyboard

Control keyboard behavior by `keyboard`.

```
import React, { useState } from 'react';
import { Checkbox, InputNumber, Space } from '

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

  return (
    <Space>
      <InputNumber min={1} max={10} keyboard={
      <Checkbox
        onChange={() => {
          setKeyboard(!keyboard);
        }}
        checked={keyboard}
      >
        Toggle keyboard
      </Checkbox>
    </Space>
  );
};

export default App;
```

$ 1,000 100%

Formatter

Display value within it's situation with `formatter`, and we usually use `parser` at the same time.

> Here is a Intl.NumberFormat InputNumber implementation: https://codesandbox.io/s/currency-wrapper-antd-input-3ynzo

```
import React from 'react';
import { InputNumber, Space } from 'antd';

const onChange = (value: number | string) => {
  console.log('changed', value);
};

const App: React.FC = () => (
  <Space>
    <InputNumber
      defaultValue={1000}
      formatter={(value) => `$ ${value}`.repl
      parser={(value) => value!.replace(/\$\s
      onChange={onChange}
    />
    <InputNumber
      defaultValue={100}
      min={0}
      max={100}
      formatter={(value) => `${value}%`}
      parser={(value) => value!.replace('%', '
      onChange={onChange}
    />
  </Space>
);

export default App;
```

3

Borderless

No border.

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

const App: React.FC = () => <InputNumber min={

export default App;
```

Out of range

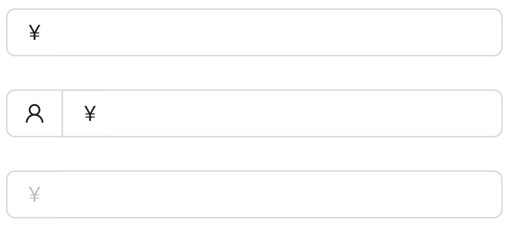
Show warning style when `value` is out of range by control.

```
import React, { useState } from 'react';
import { Button, InputNumber, Space } from 'ar

const App: React.FC = () => {
  const [value, setValue] = useState<string |

  return (
    <Space>
      <InputNumber min={1} max={10} value={val
      <Button
        type="primary"
        onClick={() => {
          setValue(99);
        }}
      >
        Reset
      </Button>
    </Space>
  );
};

export default App;
```



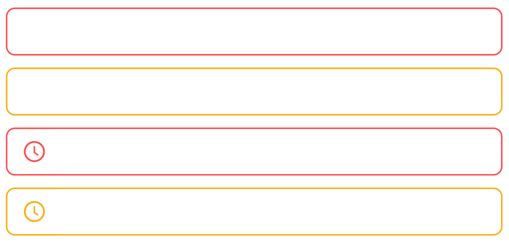
Prefix

Add a prefix inside input.

```
import React from 'react';
import { UserOutlined } from '@ant-design/icor
import { InputNumber } from 'antd';

const App: React.FC = () => (
  <>
    <InputNumber prefix="￥" style={{ width: '
    <br />
    <br />
    <InputNumber addonBefore={<UserOutlined /:
    <br />
    <br />
    <InputNumber prefix="￥" disabled style={{
  </>
);

export default App;
```



Status

Add status to InputNumber with `status`, which could be `error` or `warning`.

```
import React from 'react';
import ClockCircleOutlined from '@ant-design/i
import { InputNumber, Space } from 'antd';

const App: React.FC = () => (
  <Space direction="vertical" style={{ width:
    <InputNumber status="error" style={{ widtl
    <InputNumber status="warning" style={{ wic
    <InputNumber status="error" style={{ widtl
    <InputNumber status="warning" style={{ wic
  </Space>
);

export default App;
```

API

Property	Description	Type	Default
addonAfter	The label text displayed after (on the right side of) the input field	ReactNode	-
addonBefore	The label text displayed before (on the left side of) the input field	ReactNode	-
autoFocus	If get focus when component mounted	boolean	false
bordered	Whether has border style	boolean	true
controls	Whether to show `+-` controls, or set custom arrows icon	boolean | { upIcon?: React.ReactNode; downIcon?: React.ReactNode; }	-
decimalSeparator	Decimal separator	string	-
defaultValue	The initial value	number	-
disabled	If disable the input	boolean	false
formatter	Specifies the format of the value presented	function(value: number | string, info: { userTyping: boolean, input: string }): string	-
keyboard	If enable keyboard behavior	boolean	true
max	The max value	number	Number.MAX_SAFE_INTEGEI
min	The min value	number	Number.MIN_SAFE_INTEGEI
parser	Specifies the value extracted from formatter	function(string): number	-
precision	The precision of input value. Will use `formatter` when config of `formatter`	number	-
readOnly	If readonly the input	boolean	false
status	Set validation status	'error' | 'warning'	-
prefix	The prefix icon for the Input	ReactNode	-
size	The height of input	`large` | `middle`	-

Property	Description	Type	Default
	box	| `small`	
step	The number to which the current value is increased or decreased. It can be an integer or decimal	number | string	1
stringMode	Set value as string to support high precision decimals. Will return string value by `onChange`	boolean	false
value	The current value	number	-
onChange	The callback triggered when the value is changed	function(value: number | string | null)	-
onPressEnter	The callback function that is triggered when Enter key is pressed	function(e)	-
onStep	The callback function that is triggered when click up or down buttons	(value: number, info: { offset: number, type: 'up' | 'down' }) => void	-

Methods

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`

Token Name	Description	Type	Default Value
colorError	Used to represent the visual elements of the operation failure, such as the error Button, error Result component, etc.	`string`	`#ff4d4f`
colorErrorBorderHover	The hover state border color of the error state.	`string`	`#ffa39e`
colorErrorOutline	Control the outline color of input component in error state.	`string`	`rgba(255, 38, 5, 0.06)`
colorFillAlter	Control the alternative background color of element.	`string`	`rgba(0, 0, 0, 0.02)`
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`
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)`
colorTextDescription	Control the font color of text description.	`string`	`rgba(0, 0, 0, 0.45)`
colorTextDisabled	Control the color of text in disabled state.	`string`	`rgba(0, 0, 0, 0.25)`
colorTextPlaceholder	Control the color of placeholder text.	`string`	`rgba(0, 0, 0, 0.25)`
colorWarning	Used to represent the warning map token, such as Notification, Alert, etc. Alert or Control component(like Input) will use these map tokens.	`string`	`#faad14`
colorWarningBorderHover	The hover state border color of the warning state.	`string`	`#ffd666`
colorWarningOutline	Control the outline color of input component in warning state.	`string`	`rgba(255, 215, 5, 0.1)`
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

Token Name	Description	Type	Default Value
controlHeight	The height of the basic controls such as buttons and input boxes in Ant Design	`number`	32
controlHeightLG	LG component height	`number`	40
controlHeightSM	SM component height	`number`	24
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
controlPaddingHorizontal	Control the horizontal padding of an element.	`number`	12
controlPaddingHorizontalSM	Control the horizontal padding of an element with a small-medium size.	`number`	8
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
lineHeightLG	Line height of large text.	`number`	1.5
lineType	Border style of base components	`string`	solid
lineWidth	Border width of base components	`number`	1
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
paddingSM	Control the small padding of the element.	`number`	12
paddingXS	Control the extra small padding of the element.	`number`	8
paddingXXS	Control the extra extra small padding of the element.	`number`	4

Notes

Per issues #21158, #17344, #9421, and documentation about inputs, it appears this community does not support native inclusion of the `type="number"` in the `<Input />` attributes, so please feel free to include it as needed, and be aware that it is heavily suggested that server side validation be utilized, as client side validation can be edited by power users.

FAQ

Why `value` can exceed `min` or `max` in control?

Developer handle data by their own in control. It will make data out of sync if InputNumber change display value. It also cause potential data issues when use in form.

Why dynamic change `min` or `max` which makes `value` out of range will not trigger `onChange`?

`onChange` is user trigger event. Auto trigger will makes form lib can not detect data modify source.