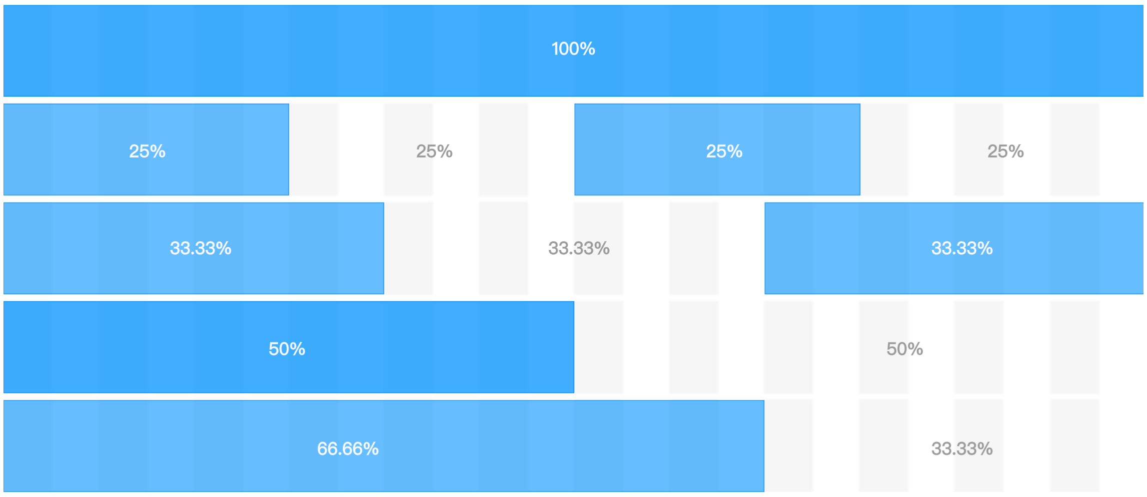
设计理念



在多数业务情况下，Ant Design 需要在设计区域内解决大量信息收纳的问题，因此在 12 栅格系统的基础上，我们将整个设计建议区域按照 24 等分的原则进行划分。

划分之后的信息区块我们称之为『盒子』。建议横向排列的盒子数量最多四个，最少一个。『盒子』在整个屏幕上占比见上图。设计部分基于盒子的单位定制盒子内部的排版规则，以保证视觉层面的舒适感。

概述

布局的栅格化系统，我们是基于行（row）和列（col）来定义信息区块的外部框架，以保证页面的每个区域能够稳健地排布起来。下面简单介绍一下它的工作原理：

- 通过 `row` 在水平方向建立一组 `column`（简写 col）。
- 你的内容应当放置于 `col` 内，并且，只有 `col` 可以作为 `row` 的直接元素。
- 栅格系统中的列是指 1 到 24 的值来表示其跨越的范围。例如，三个等宽的列可以使用 `<Col span={8} />` 来创建。
- 如果一个 `row` 中的 `col` 总和超过 24，那么多余的 `col` 会作为一个整体另起一行排列。

我们的栅格化系统基于 Flex 布局，允许子元素在父节点内的水平对齐方式 - 居左、居中、居右、等宽排列、分散排列。子元素与子元素之间，支持顶部对齐、垂直居中对齐、底部对齐的方式。同时，支持使用 order 来定义元素的排列顺序。

布局是基于 24 栅格来定义每一个『盒子』的宽度，但不拘泥于栅格。

代码演示

基础栅格

```
import React from 'react';
import { Col, Row } from 'antd';

const App: React.FC = () => (
  <>
```

```
    <Row>
      <Col span={24}>col</Col>
    </Row>
    <Row>
      <Col span={12}>col-12</Col>
      <Col span={12}>col-12</Col>
    </Row>
    <Row>
      <Col span={8}>col-8</Col>
      <Col span={8}>col-8</Col>
      <Col span={8}>col-8</Col>
    </Row>
    <Row>
      <Col span={6}>col-6</Col>
      <Col span={6}>col-6</Col>
      <Col span={6}>col-6</Col>
      <Col span={6}>col-6</Col>
    </Row>
  </>
);

export default App;
```

区块间隔

```
import React from 'react';
import { Col, Divider, Row } from 'antd';

const style: React.CSSProperties = { background: '#0092ff', padding: '8px
0' };

const App: React.FC = () => (
  <>
    <Divider orientation="left">Horizontal</Divider>
    <Row gutter={16}>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
    </Row>
    <Divider orientation="left">Responsive</Divider>
    <Row gutter={{ xs: 8, sm: 16, md: 24, lg: 32 }}>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
    </Row>
    <Divider orientation="left">Vertical</Divider>
    <Row gutter={[16, 24]}>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
      <Col className="gutter-row" span={6}>
        <div style={style}>col-6</div>
      </Col>
    </Row>
  </>
);
```

```
export default App;
```

左右偏移

```
import React from 'react';
import { Col, Row } from 'antd';

const App: React.FC = () => (
  <>
    <Row>
      <Col span={8}>col-8</Col>
      <Col span={8} offset={8}>
        col-8
      </Col>
    </Row>
    <Row>
      <Col span={6} offset={6}>
        col-6 col-offset-6
      </Col>
      <Col span={6} offset={6}>
        col-6 col-offset-6
      </Col>
    </Row>
    <Row>
      <Col span={12} offset={6}>
        col-12 col-offset-6
      </Col>
    </Row>
  </>
);

export default App;
```

栅格排序

```
import React from 'react';
import { Col, Row } from 'antd';

const App: React.FC = () => (
  <Row>
    <Col span={18} push={6}>
      col-18 col-push-6
    </Col>
    <Col span={6} pull={18}>
```

```
      col-6 col-pull-18
    </Col>
  </Row>
);

export default App;
```

排版

```
import React from 'react';
import { Col, Divider, Row } from 'antd';

const App: React.FC = () => (
  <>
    <Divider orientation="left">sub-element align left</Divider>
    <Row justify="start">
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
    </Row>

    <Divider orientation="left">sub-element align center</Divider>
    <Row justify="center">
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
    </Row>

    <Divider orientation="left">sub-element align right</Divider>
    <Row justify="end">
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
    </Row>

    <Divider orientation="left">sub-element monospaced
arrangement</Divider>
    <Row justify="space-between">
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
    </Row>
    <Divider orientation="left">sub-element align full</Divider>
    <Row justify="space-around">
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
    </Row>

    <Divider orientation="left">sub-element align evenly</Divider>
    <Row justify="space-evenly">
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
      <Col span={4}>col-4</Col>
    </Row>
  </>
);

export default App;
```

对齐

```
import React from 'react';
import { Col, Divider, Row } from 'antd';

const DemoBox: React.FC<React.PropsWithChildren<{ value: number }>> =
(props) => (
  <p className={`height-${props.value}`}>{props.children}</p>
);

const App: React.FC = () => (
  <>
    <Divider orientation="left">Align Top</Divider>
    <Row justify="center" align="top">
      <Col span={4}>
        <DemoBox value={100}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={50}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={120}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={80}>col-4</DemoBox>
      </Col>
    </Row>

    <Divider orientation="left">Align Middle</Divider>
    <Row justify="space-around" align="middle">
      <Col span={4}>
        <DemoBox value={100}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={50}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={120}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={80}>col-4</DemoBox>
      </Col>
    </Row>

    <Divider orientation="left">Align Bottom</Divider>
    <Row justify="space-between" align="bottom">
      <Col span={4}>
        <DemoBox value={100}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={50}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={120}>col-4</DemoBox>
      </Col>
      <Col span={4}>
        <DemoBox value={80}>col-4</DemoBox>
      </Col>
    </Row>
  </>
);

export default App;
```

排序

```
import React from 'react';
import { Col, Divider, Row } from 'antd';

const App: React.FC = () => (
  <>
```

```
    <Divider orientation="left">Normal</Divider>
    <Row>
      <Col span={6} order={4}>
        1 col-order-4
      </Col>
      <Col span={6} order={3}>
        2 col-order-3
      </Col>
      <Col span={6} order={2}>
        3 col-order-2
      </Col>
      <Col span={6} order={1}>
        4 col-order-1
      </Col>
    </Row>
    <Divider orientation="left">Responsive</Divider>
    <Row>
      <Col span={6} xs={{ order: 1 }} sm={{ order: 2 }} md={{ order: 3 }}
lg={{ order: 4 }}>
        1 col-order-responsive
      </Col>
      <Col span={6} xs={{ order: 2 }} sm={{ order: 1 }} md={{ order: 4 }}
lg={{ order: 3 }}>
        2 col-order-responsive
      </Col>
      <Col span={6} xs={{ order: 3 }} sm={{ order: 4 }} md={{ order: 2 }}
lg={{ order: 1 }}>
        3 col-order-responsive
      </Col>
      <Col span={6} xs={{ order: 4 }} sm={{ order: 3 }} md={{ order: 1 }}
lg={{ order: 2 }}>
        4 col-order-responsive
      </Col>
    </Row>
  </>
);

export default App;
```

Flex 填充

```
import React from 'react';
import { Col, Divider, Row } from 'antd';

const App: React.FC = () => (
  <>
```

```
    <Divider orientation="left">Percentage columns</Divider>
    <Row>
      <Col flex={2}>2 / 5</Col>
      <Col flex={3}>3 / 5</Col>
    </Row>
    <Divider orientation="left">Fill rest</Divider>
    <Row>
      <Col flex="100px">100px</Col>
      <Col flex="auto">Fill Rest</Col>
    </Row>
    <Divider orientation="left">Raw flex style</Divider>
    <Row>
      <Col flex="1 1 200px">1 1 200px</Col>
      <Col flex="0 1 300px">0 1 300px</Col>
    </Row>

    <Row wrap={false}>
      <Col flex="none">
        <div style={{ padding: '0 16px' }}>none</div>
      </Col>
      <Col flex="auto">auto with no-wrap</Col>
    </Row>
  </>
);

export default App;
```

响应式布局

```
import React from 'react';
import { Col, Row } from 'antd';

const App: React.FC = () => (
  <Row>
    <Col xs={2} sm={4} md={6} lg={8} xl={10}>
      Col
    </Col>
    <Col xs={20} sm={16} md={12} lg={8} xl={4}>
      Col
    </Col>
    <Col xs={2} sm={4} md={6} lg={8} xl={10}>
      Col
    </Col>
  </Row>
);
```

```
export default App;
```

Flex 响应式布局

v5.14.0

```
import React from 'react';
import { Col, Row } from 'antd';

const App: React.FC = () => (
  <Row>
    {Array.from({ length: 10 }).map((_, index) => {
      const key = `col-${index}`;
      return (
        <Col
          key={key}
          xs={{ flex: '100%' }}
          sm={{ flex: '50%' }}
          md={{ flex: '40%' }}
          lg={{ flex: '20%' }}
          xl={{ flex: '10%' }}
        >
          Col
        </Col>
      );
    })}
  </Row>
);

export default App;
```

其他属性的响应式

```
import React from 'react';
import { Col, Row } from 'antd';

const App: React.FC = () => (
  <Row>
    <Col xs={{ span: 5, offset: 1 }} lg={{ span: 6, offset: 2 }}>
      Col
    </Col>
    <Col xs={{ span: 11, offset: 1 }} lg={{ span: 6, offset: 2 }}>
      Col
    </Col>
    <Col xs={{ span: 5, offset: 1 }} lg={{ span: 6, offset: 2 }}>
```

```
      Col
    </Col>
  </Row>
);

export default App;
```

栅格配置器

```
import React, { useState } from 'react';
import { Col, Row, Slider } from 'antd';

const gutters: Record<PropertyKey, number> = {};
const vgutters: Record<PropertyKey, number> = {};
const colCounts: Record<PropertyKey, number> = {};

[8, 16, 24, 32, 40, 48].forEach((value, i) => {
  gutters[i] = value;
});
[8, 16, 24, 32, 40, 48].forEach((value, i) => {
  vgutters[i] = value;
});
[2, 3, 4, 6, 8, 12].forEach((value, i) => {
  colCounts[i] = value;
});

const App: React.FC = () => {
  const [gutterKey, setGutterKey] = useState(1);
  const [vgutterKey, setVgutterKey] = useState(1);
  const [colCountKey, setColCountKey] = useState(2);

  const cols = [];
  const colCount = colCounts[colCountKey];
  let colCode = '';
  for (let i = 0; i < colCount; i++) {
    cols.push(
      <Col key={i.toString()} span={24 / colCount}>
        <div>Column</div>
      </Col>,
    );
    colCode += `  <Col span={${24 / colCount}} />\n`;
  }

  return (
    <>
      <span>Horizontal Gutter (px): </span>
      <div style={{ width: '50%' }}>
        <Slider
          min={0}
          max={Object.keys(gutters).length - 1}
          value={gutterKey}
          onChange={setGutterKey}
          marks={gutters}
          step={null}
          tooltip={{ formatter: (value) => gutters[value as number] }}
        />
      </div>
      <span>Vertical Gutter (px): </span>
      <div style={{ width: '50%' }}>
        <Slider
          min={0}
          max={Object.keys(vgutters).length - 1}
          value={vgutterKey}
          onChange={setVgutterKey}
          marks={vgutters}
          step={null}
          tooltip={{ formatter: (value) => vgutters[value as number] }}
        />
      </div>
      <span>Column Count:</span>
      <div style={{ width: '50%', marginBottom: 48 }}>
        <Slider
          min={0}
          max={Object.keys(colCounts).length - 1}
          value={colCountKey}
          onChange={setColCountKey}
          marks={colCounts}
          step={null}
          tooltip={{ formatter: (value) => colCounts[value as number] }}
        />
      </div>
      <Row gutter={[gutters[gutterKey], vgutters[vgutterKey]]}>
        {cols}
        {cols}
      </Row>
      Another Row:
      <Row gutter={[gutters[gutterKey], vgutters[vgutterKey]]}>{cols}</Row>
      <pre className="demo-code">{`<Row gutter={[${gutters[gutterKey]},
${vgutters[vgutterKey]}]}>\n${colCode}\n${colCode}</Row>`}</pre>
      <pre className="demo-code">{`<Row gutter={[${gutters[gutterKey]},
${vgutters[vgutterKey]}]}>\n${colCode}</Row>`}</pre>
    </>
```

```
  );
};

export default App;
```

useBreakpoint Hook

```
import React from 'react';
import { Grid, Tag } from 'antd';

const { useBreakpoint } = Grid;

const App: React.FC = () => {
  const screens = useBreakpoint();

  return (
    <>
      Current break point:{' '}
      {Object.entries(screens)
        .filter((screen) => !!screen[1])
        .map((screen) => (
          <Tag color="blue" key={screen[0]}>
            {screen[0]}
          </Tag>
        ))}
    </>
  );
};

export default App;
```

API

通用属性参考：[通用属性]

Ant Design 的布局组件若不能满足你的需求，你也可以直接使用社区的优秀布局组件：

- [react-flexbox-grid]
- [react-blocks]

Row

参数	说明	类型	默认值	版本
align	垂直对齐方式	top | middle | bottom | stretch | `{[key in 'xs' | 'sm' | 'md' | 'lg' | 'xl' | 'xxl']: 'top' |`	top	object: 4.24.0

		'middle' | 'bottom' | 'stretch'}		
gutter	栅格间隔，可以写成像素值或支持响应式的对象写法来设置水平间隔 { xs: 8, sm: 16, md: 24}。或者使用数组形式同时设置 [水平间距, 垂直间距]	number | object | array	0	
justify	水平排列方式	start | end | center | space-around | space-between | space-evenly | {[key in 'xs' | 'sm' | 'md' | 'lg' | 'xl' | 'xxl']: 'start' | 'end' | 'center' | 'space-around' | 'space-between' | 'space-evenly'}	start	object: 4.24.0
wrap	是否自动换行	boolean	true	4.8.0

Col

参数	说明	类型	默认值	版本
flex	flex 布局属性	string | number	-	
offset	栅格左侧的间隔格数，间隔内不可以有栅格	number	0	
order	栅格顺序	number	0	
pull	栅格向左移动格数	number	0	
push	栅格向右移动格数	number	0	
span	栅格占位格数，为 0 时相当于 `display: none`	number	-	
xs	窗口宽度 < 576px 响应式栅格，可为栅格数或一个包含其他属性的对象	number | object	-	
sm	窗口宽度 ≥ 576px 响应式栅格，可为栅格数或一个包含其他属性的对象	number | object	-	
md	窗口宽度 ≥ 768px 响应式栅格，可为栅格数或一个包含其他属性的对象	number | object	-	

lg	窗口宽度 `≥ 992px` 响应式栅格，可为栅格数或一个包含其他属性的对象	number | object	-	
xl	窗口宽度 `≥ 1200px` 响应式栅格，可为栅格数或一个包含其他属性的对象	number | object	-	
xxl	窗口宽度 `≥ 1600px` 响应式栅格，可为栅格数或一个包含其他属性的对象	number | object	-	

您可以使用 主题定制 修改 `screen[XS|SM|MD|LG|XL|XXL]` 来修改断点值（自 5.1.0 起，codesandbox demo）。

响应式栅格的断点扩展自 BootStrap 4 的规则（不包含链接里 `occasionally` 的部分)。

主题变量（Design Token）