Stack

Stack is a container component for arranging elements vertically or horizontally.

Introduction

The Stack component manages the layout of its immediate children along the vertical or horizontal axis, with optional spacing and dividers between each child.

> ⓘ Stack is ideal for one-dimensional layouts, while Grid is preferable when you need both vertical *and* horizontal arrangement.

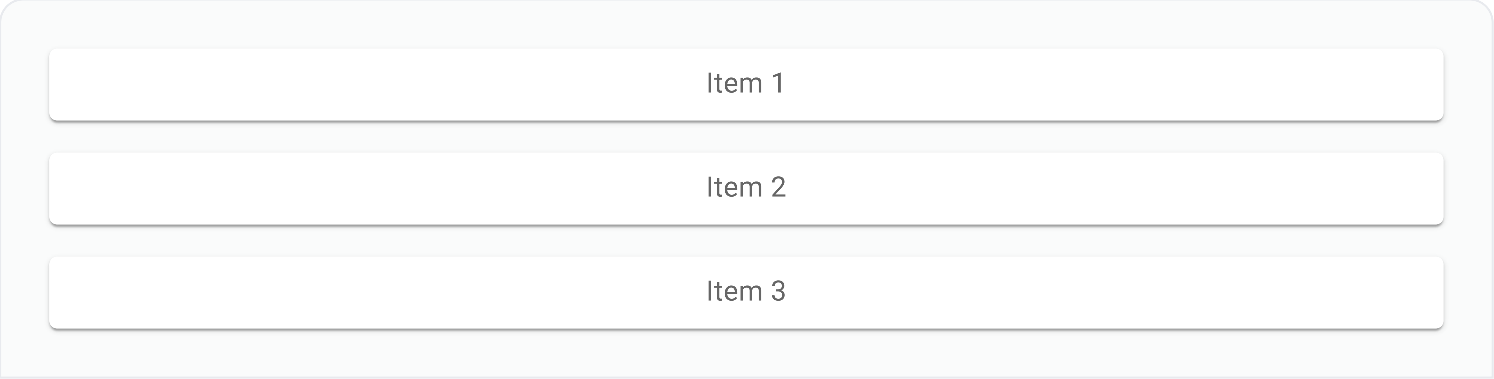
View as Markdown Feedback Bundle size Source Figma Sketch

Basics

```
import Stack from '@mui/material/Stack';
```

The Stack component acts as a generic container, wrapping around the elements to be arranged.

Use the `spacing` prop to control the space between children. The spacing value can be any number, including decimals, or a string. (The prop is converted into a CSS property using the [`theme.spacing()`](#) helper.)



Item 1

Item 2

Item 3

Edit in Chat JS TS Collapse code

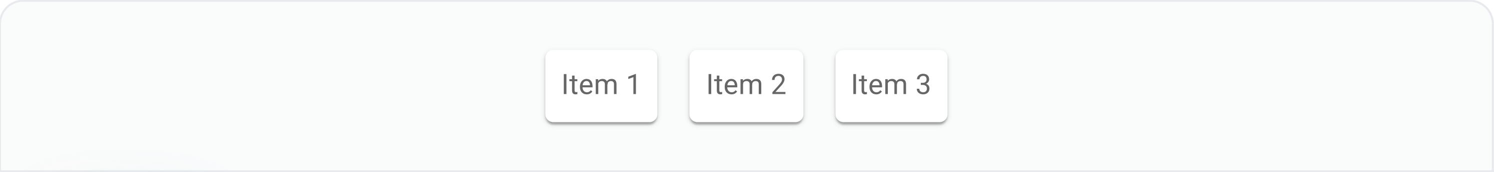
```
import Box from '@mui/material/Box';
import Paper from '@mui/material/Paper';
import Stack from '@mui/material/Stack';
import { styled } from '@mui/material/styles';

const Item = styled(Paper)(({ theme }) => ({
  backgroundColor: '#fff',
  ...theme.typography.body2,
  padding: theme.spacing(1),
  textAlign: 'center',
  color: (theme.vars ?? theme).palette.text.secondary,
  ...theme.applyStyles('dark', {
    backgroundColor: '#1A2027',
  }),
}));

export default function BasicStack() {
  return (
    <Box sx={{ width: '100%' }}>
      <Stack spacing={2}>
        <Item>Item 1</Item>
        <Item>Item 2</Item>
        <Item>Item 3</Item>
      </Stack>
    </Box>
  );
}
```

Stack vs. Grid

`Stack` is concerned with one-dimensional layouts, while Grid handles two-dimensional layouts. The default direction is `column` which stacks children vertically.

Direction

By default, Stack arranges items vertically in a column. Use the `direction` prop to position items horizontally in a row:

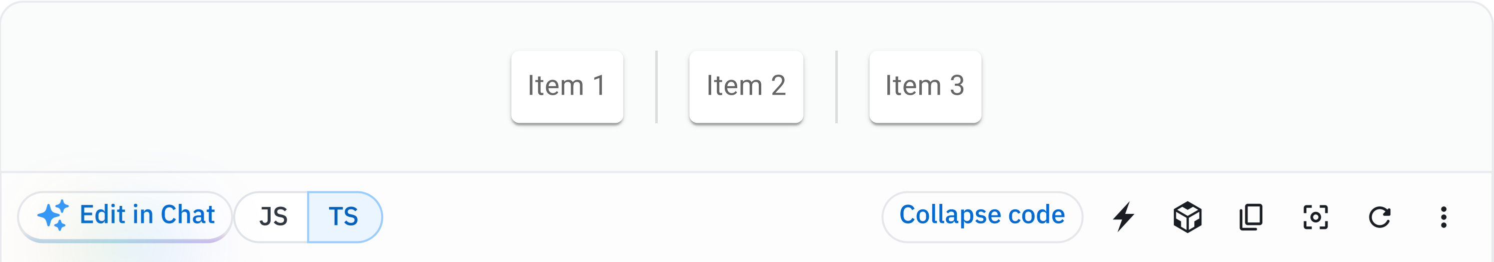


Edit in Chat JS TS Collapse code

```
import Paper from '@mui/material/Paper';
import Stack from '@mui/material/Stack';
import { styled } from '@mui/material/styles';

const Item = styled(Paper)(({ theme }) => ({
  backgroundColor: '#fff',
  ...theme.typography.body2,
  padding: theme.spacing(1),
  textAlign: 'center',
  color: (theme.vars ?? theme).palette.text.secondary,
  ...theme.applyStyles('dark', {
    backgroundColor: '#1A2027',
  }),
}));

export default function DirectionStack() {
  return (
    <div>
      <Stack direction="row" spacing={2}>
        <Item>Item 1</Item>
        <Item>Item 2</Item>
        <Item>Item 3</Item>
      </Stack>
    </div>
  );
}
```

Dividers

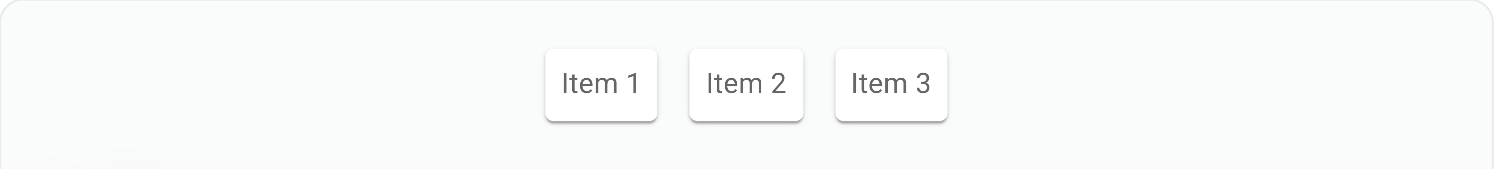
Use the `divider` prop to insert an element between each child. This works particularly well with the [Divider](#) component, as shown below:



```
import Divider from '@mui/material/Divider';
import Paper from '@mui/material/Paper';
import Stack from '@mui/material/Stack';
import { styled } from '@mui/material/styles';

const Item = styled(Paper)(({ theme }) => ({
  backgroundColor: '#fff',
  ...theme.typography.body2,
  padding: theme.spacing(1),
  textAlign: 'center',
  color: (theme.vars ?? theme).palette.text.secondary,
  ...theme.applyStyles('dark', {
    backgroundColor: '#1A2027',
  }),
}));

export default function DividerStack() {
  return (
    <div>
      <Stack
        direction="row"
        divider={<Divider orientation="vertical" flexItem />}
        spacing={2}
      >
        <Item>Item 1</Item>
        <Item>Item 2</Item>
        <Item>Item 3</Item>
      </Stack>
    </div>
  );
}
```

Responsive values

You can switch the `direction` or `spacing` values based on the active breakpoint.



Edit in Chat JS TS Collapse code

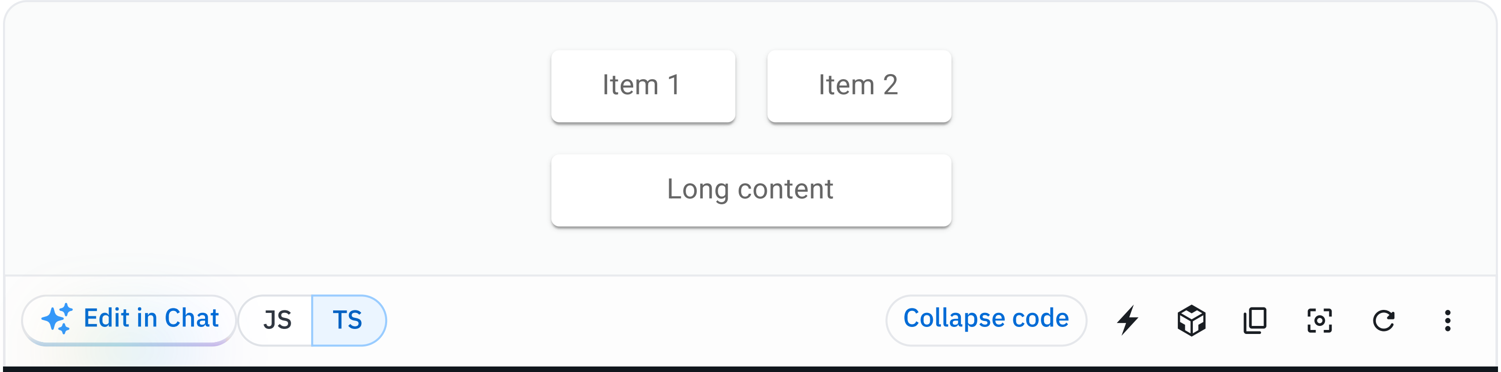
```
import Paper from '@mui/material/Paper';
import Stack from '@mui/material/Stack';
import { styled } from '@mui/material/styles';

const Item = styled(Paper)(({ theme }) => ({
  backgroundColor: '#fff',
  ...theme.typography.body2,
  padding: theme.spacing(1),
  textAlign: 'center',
  color: (theme.vars ?? theme).palette.text.secondary,
  ...theme.applyStyles('dark', {
    backgroundColor: '#1A2027',
  }),
}));

export default function ResponsiveStack() {
  return (
    <div>
      <Stack
        direction={{ xs: 'column', sm: 'row' }}
        spacing={{ xs: 1, sm: 2, md: 4 }}
      >
        <Item>Item 1</Item>
        <Item>Item 2</Item>
        <Item>Item 3</Item>
      </Stack>
    </div>
  );
}
```

Flexbox gap

To use flexbox `gap` for the spacing implementation, set the `useFlexGap` prop to true.

It removes the known limitations of the default implementation that uses CSS nested selector. However, CSS flexbox gap is not fully supported in some browsers.

We recommend checking the support percentage before using it.



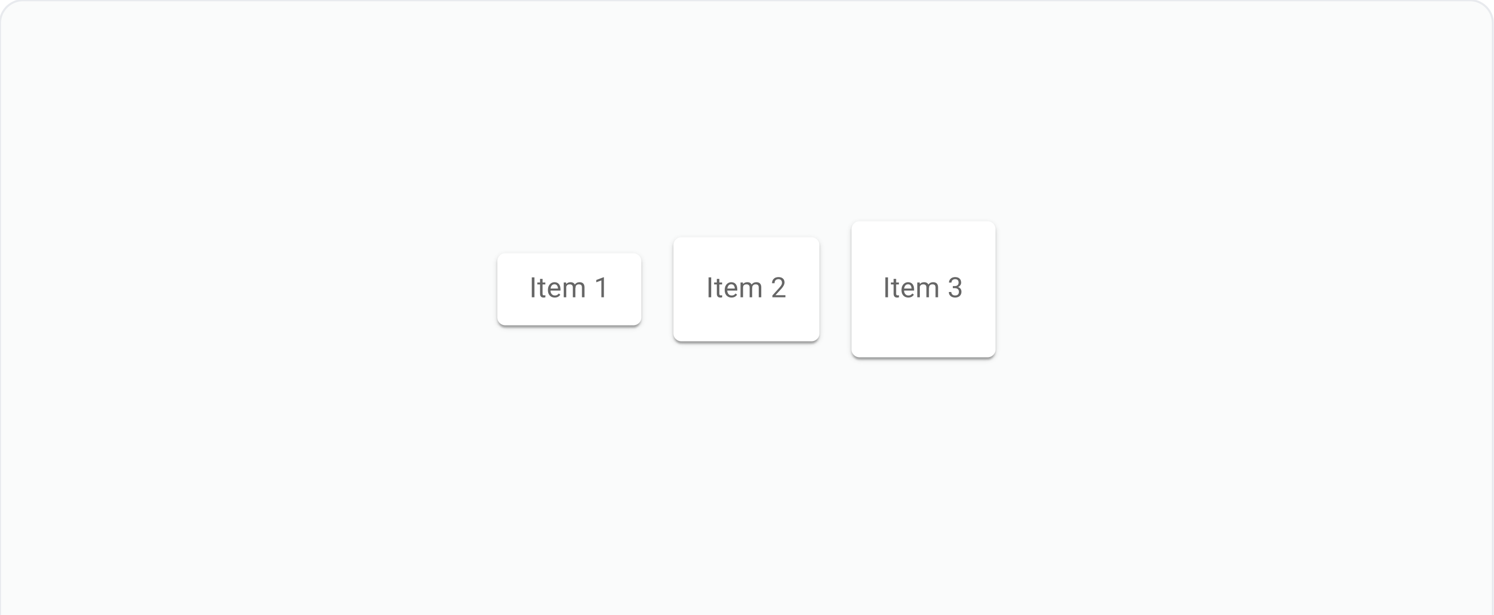
```
import Paper from '@mui/material/Paper';
import Stack from '@mui/material/Stack';
import Box from '@mui/material/Box';
import { styled } from '@mui/material/styles';

const Item = styled(Paper)(({ theme }) => ({
  backgroundColor: '#fff',
  ...theme.typography.body2,
  padding: theme.spacing(1),
  textAlign: 'center',
  color: (theme.vars ?? theme).palette.text.secondary,
  flexGrow: 1,
  ...theme.applyStyles('dark', {
    backgroundColor: '#1A2027',
  }),
}));

export default function FlexboxGapStack() {
  return (
    <Box sx={{ width: 200 }}>
      <Stack
        spacing={{ xs: 1, sm: 2 }}
        direction="row"
        useFlexGap
        sx={{ flexWrap: 'wrap' }}
      >
        <Item>Item 1</Item>
        <Item>Item 2</Item>
        <Item>Long content</Item>
      </Stack>
    </Box>
  );
}
```

To set the prop to all stack instances, create a theme with default props:

```
import { ThemeProvider, createTheme } from '@mui/material/styles';
import Stack from '@mui/material/Stack';

const theme = createTheme({
  components: {
    MuiStack: {
      defaultProps: {
        useFlexGap: true,
      },
    },
  },
});

function App() {
  return (
    <ThemeProvider theme={theme}>
      <Stack>…</Stack> {/* uses flexbox gap by default */}
    </ThemeProvider>
  );
}
```

Interactive demo

Below is an interactive demo that lets you explore the visual results of the different settings:



direction

● row ○ row-reverse ○ column ○ column-reverse

alignItems

○ flex-start ● center ○ flex-end ○ stretch ○ baseline

justifyContent

○ flex-start ● center ○ flex-end ○ space-between ○ space-around

○ space-evenly

spacing

○ 0 ○ 0.5 ○ 1 ● 2 ○ 3 ○ 4 ○ 8 ○ 12

```
<Stack
  direction="row"
  spacing={2}
  sx={{
    justifyContent: "center",
    alignItems: "center",
  }}
>
```

Copy

System props

System props are deprecated and will be removed in the next major release. Please use the `sx` prop instead.

```
- <Stack mt={2} />
+ <Stack sx={{ mt: 2 }} />
```

Copy

Limitations

Margin on the children

Customizing the margin on the children is not supported by default.

For instance, the top-margin on the `Button` component below will be ignored.

```
<Stack>
  <Button sx={{ marginTop: '30px' }}>...</Button>
</Stack>
```

To overcome this limitation, set `useFlexGap` prop to true to switch to CSS flexbox gap implementation.

You can learn more about this limitation by visiting this RFC.

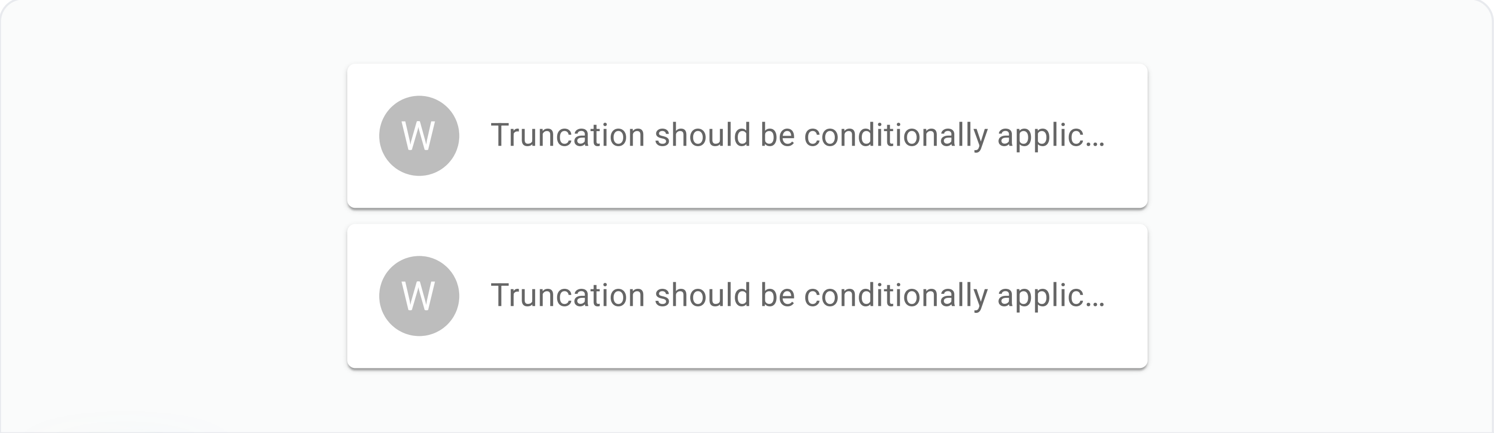
white-space: nowrap

The initial setting on flex items is `min-width: auto`. This causes a positioning conflict when children use `white-space: nowrap;`. You can reproduce the issue with:

```
<Stack direction="row">
  <Typography noWrap>
```

In order for the item to stay within the container you need to set `min-width: 0`.

```
<Stack direction="row" sx={{ minWidth: 0 }}>
  <Typography noWrap>
```



W Truncation should be conditionally applic…

W Truncation should be conditionally applic…

Edit in Chat JS TS Collapse code

```
import Avatar from '@mui/material/Avatar';
import Box from '@mui/material/Box';
import Paper from '@mui/material/Paper';
import Stack from '@mui/material/Stack';
import { styled } from '@mui/material/styles';
import Typography from '@mui/material/Typography';

const Item = styled(Paper)(({ theme }) => ({
  backgroundColor: '#fff',
  ...theme.typography.body2,
  padding: theme.spacing(1),
  textAlign: 'center',
  color: (theme.vars ?? theme).palette.text.secondary,
  maxWidth: 400,
  ...theme.applyStyles('dark', {
    backgroundColor: '#1A2027',
  }),
}));

const message = `Truncation should be conditionally applicable on this long line of text
 as this is a much longer line than what the container can support.`;

export default function ZeroWidthStack() {
  return (
    <Box sx={{ flexGrow: 1, overflow: 'hidden', px: 3 }}>
      <Item sx={{ my: 1, mx: 'auto', p: 2 }}>
        <Stack spacing={2} direction="row" sx={{ alignItems: 'center' }}>
          <Avatar>W</Avatar>
          <Typography noWrap>{message}</Typography>
        </Stack>
      </Item>
      <Item sx={{ my: 1, mx: 'auto', p: 2 }}>
        <Stack spacing={2} direction="row" sx={{ alignItems: 'center' }}>
          <Stack>
            <Avatar>W</Avatar>
          </Stack>
          <Stack sx={{ minWidth: 0 }}>
            <Typography noWrap>{message}</Typography>
```

Anatomy

The Stack component is composed of a single root `<div>` element:

```
<div class="MuiStack-root">
  <!-- Stack contents -->
</div>
```

API

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

- `<PigmentStack />`
- `<Stack />`

Edit this page

Was this page helpful?

‹ GridLegacy

Image List ›