**Sistema grid Bootstrap**

Use nosso poderoso, mobile-first, grid flexbox para fazer layouts de todas as formas e tamanhos, graças ao sistema de doze colunas, cinco breakpoints responsivos, variáveis e mixins Sass, além de várias classes pré-definidas.

Como funciona

O sistema grid Bootstrap usa vários containers, linhas e colunas para arranjar e alinhar conteúdo. Ele é feito com flexbox e é, totalmente, responsivo. Abaixo, temos um exemplo e aprofundamento sobre como o grid funciona.

**Novo ou não sabe muito sobre flexbox?** Leia este guia flexbox, do CSS Tricks, para background, terminologia, diretrizes e snippets.

Uma de três colunas

Uma de três colunas

Uma de três colunas

Copy

<div class="container">

<div class="row">

<div class="col-sm">

Uma de três colunas

</div>

<div class="col-sm">

Uma de três colunas

</div>

<div class="col-sm">

Uma de três colunas

</div>

</div>

</div>

O exemplo acima cria três colunas de larguras idênticas em dispositivos pequenos, médios, grandes e extra grandes, usando nossas classes grid pré-definidas. Estas colunas são centralizadas na página, usando o elemento pai .container.

Resumindo, assim é como funciona:

* Containers criam meios para centralizar e, horizontalmente, preencher os conteúdos de seu site;
  + Use .container para ter uma largura responsiva em pixel ou .container-fluid para ter width: 100%, em todas viewports e tamanhos de disposivos.
* Rows são elemntos para envolver colunas;
  + Cada coluna tem padding horizontal (gutter) para controlar o espaço, entre elas.
    - Este padding, depois, é cancelado com rows usando margens negativas. Assim, todo conteúdo em suas colunas é, visualmente, alinhado a esquerda.
* Em um layout grid, o conteúdo deve ser posicionado dentro de colunas e só elas podem ser filhos imediatos de .rows;
* Graças ao flexbox, colunas grid sem width declarado vão, automaticamente, se dimensionar com larguras iguais;
  + Por exemplo, quatro exemplos de .col-sm vão, automaticamente, ter 25% de largura, no breakpoint sm ou maior;
  + Veja a seção colunas com layout automático, para mais exemplos.
* Classes de colunas indicam o número de colunas que você quer usar, dentro de uma possibilidade de 12, por row;
  + Se você quiser três colunas de larguras idênticas, pode usar .col-4, por exemplo.
* Largura de colunas são definidas em porcentagem para que, então, elas sejam sempre fluidas e dimensionadas com relação a seus elementos pais;
* Colunas possuem padding horizontal para criar gutters, entre cada coluna.
  + No entanto, você pode remover a margin das rows e padding das colunas, usando .no-gutters na .row.
* Para fazer o grid responsivo, existem cinco breakpoints, um para cada breakpoint responsivo: extra small (implícito), small, medium, large e extra large;
* Breakpoints grid são baseados em media queries min-width, significando que **elas aplicam estilos para o dado breakpoint e outros maiores**;
  + Exemplo: .col-sm-4 aplica ao small, medium, large e extra large, mas não ao primeiro breakpoint xs.
* Você pode usar classes grid pré-definidas (como .col-4) ou mixins Sass com uma marcação mais semântica.

Se atente as limitações e bugs flexbox, tipo a impossibilidade de usar alguns elementos HTML como container flex.

Parâmetros grid

Enquanto Bootstrap usa em ou rem para definir a maioria das dimensões, px é usado para breakpoints grid e largura de containers. Isto se deve a largura da viewport ser em pixels e não se alterar com mudança no tamanho de fonte.

Veja como aspectos do sistema grid Bootstrap funciona, através de diferentes dispositivos, numa bela tabela.

|  | **Extra small** <576px | **Small** ≥576px | **Medium** ≥768px | **Large** ≥992px | **Extra large** ≥1200px |
| --- | --- | --- | --- | --- | --- |
| **Largura máxima do container** | Não tem (auto) | 540px | 720px | 960px | 1140px |
| **Prefixo em classe** | .col- | .col-sm- | .col-md- | .col-lg- | .col-xl- |
| **Número de colunas** | 12 | | | | |
| **Largura da gutter** | 30px (15px em cada lado da coluna) | | | | |
| **Aninhável** | Sim | | | | |
| **Ordenamento de coluna** | Sim | | | | |

Layout automático de colunas

Use classes de breakpoints específicos para dimensionamento de colunas fácil, sem uma classe numerada explícita, como .col-sm-6.

Largura idêntica

Por exemplo, aqui estão dois layouts grid que aparecem em todo dispositivo e viewport, desde xs até xl. Use qualquer quantidade de classes sem números, para cada breakpoint necessário e, então, todas colunas terão a mesma largura.

1 de 2

2 de 2

1 de 3

2 de 3

3 de 3

Copy

<div class="container">

<div class="row">

<div class="col">

1 de 2

</div>

<div class="col">

2 de 2

</div>

</div>

<div class="row">

<div class="col">

1 de 3

</div>

<div class="col">

2 de 3

</div>

<div class="col">

3 de 3

</div>

</div>

</div>

Colunas com larguras idênticas podem ser quebradas em diversas linhas, mas tem um bug flexbox do Safari que evita isso de funcionar, sem flex-basis ou border declarado. Há soluções para versões antigas de browsers, mas não são necessárias, se você estiver atualizado.

Coluna

Coluna

Coluna

Coluna

Copy

<div class="container">

<div class="row">

<div class="col">Coluna</div>

<div class="col">Coluna</div>

<div class="w-100"></div>

<div class="col">Coluna</div>

<div class="col">Coluna</div>

</div>

</div>

Definindo largura de uma coluna

Layout automático em colunas flexbox do grid também significa que você pode definir a largura de uma coluna e ter suas colunas irmãs redimensionadas, automaticamente. Você também pode usar classes grid pré-definidas (como mostrado abaixo), mixins grid ou larguras inline. Perceba que as outras colunas vão se redimensionar, não importa a largura da coluna do meio.

1 de 3

2 de 3 (maior)

3 de 3

1 de 3

2 de 3 (maior)

3 de 3

Copy

<div class="container">

<div class="row">

<div class="col">

1 de 3

</div>

<div class="col-6">

2 de 3 (maior)

</div>

<div class="col">

3 de 3

</div>

</div>

<div class="row">

<div class="col">

1 de 3

</div>

<div class="col-5">

2 de 3 (maior)

</div>

<div class="col">

3 de 3

</div>

</div>

</div>

Conteúdo com largura variável

Use classes col-{breakpoint}-auto para dimensionar colunas, baseando-se na largura natural de seus conteúdos.

1 de 3

Conteúdo com largura variável

3 de 3

1 de 3

Conteúdo com largura variável

3 de 3

Copy

<div class="container">

<div class="row justify-content-md-center">

<div class="col col-lg-2">

1 de 3

</div>

<div class="col-md-auto">

Conteúdo com largura variável

</div>

<div class="col col-lg-2">

3 de 3

</div>

</div>

<div class="row">

<div class="col">

1 de 3

</div>

<div class="col-md-auto">

Conteúdo com largura variável

</div>

<div class="col col-lg-2">

3 de 3

</div>

</div>

</div>

Larguras idênticas em várias linhas

Crie colunas de larguras idênticas que se extendem por várias linhas, colocando .w-100 onde você quer que as colunas quebrem, em uma nova linha. Faça com que as quebras sejam responsivas, misturando .w-100 com algum utilitário de display responsivo.

col

col

col

col

Copy

<div class="row">

<div class="col">col</div>

<div class="col">col</div>

<div class="w-100"></div>

<div class="col">col</div>

<div class="col">col</div>

</div>

Classes responsivas

O grid Bootstrap possui cinco graus de classes pré-definidas, para construir layouts responsivos complexos. Customize o tamanho de suas colunas em dispositivos extra small, small, medium, large ou extra large.

Todos breakpoints

Para grids que são iguais, independente do tamanho dox dispositivos, use as classes .col e .col-\*. Especifique uma classe numerada quando você precisar que uma coluna específica seja dimensionada. Caso contrário, sinta-se a vontade de usar só .col.

col

col

col

col

col-8

col-4

Copy

<div class="row">

<div class="col">col</div>

<div class="col">col</div>

<div class="col">col</div>

<div class="col">col</div>

</div>

<div class="row">

<div class="col-8">col-8</div>

<div class="col-4">col-4</div>

</div>

De vertical para horizontal

Usando só um conjunto de classes .col-sm-\*, você pode criar um sistema de grid básico que inicia empilhado e termina na horizontal, a partir do breakpoint small (sm).

col-sm-8

col-sm-4

col-sm

col-sm

col-sm

Copy

<div class="row">

<div class="col-sm-8">col-sm-8</div>

<div class="col-sm-4">col-sm-4</div>

</div>

<div class="row">

<div class="col-sm">col-sm</div>

<div class="col-sm">col-sm</div>

<div class="col-sm">col-sm</div>

</div>

Misture e combine

Não quer que suas colunas, simplesmente, empilhem-se em alguns breakpoints? Use uma combinação de diferentes classes, para cada breakpoint necessário. Veja o exemplo abaixo, para ter uma melhor ideia de como isso tudo funciona.

.col-12 .col-md-8

.col-6 .col-md-4

.col-6 .col-md-4

.col-6 .col-md-4

.col-6 .col-md-4

.col-6

.col-6

Copy

<!-- Empilhe as colunas, em dispositivos móveis, fazendo com que uma tenha largura total e a outra só metade -->

<div class="row">

<div class="col-12 col-md-8">.col-12 .col-md-8</div>

<div class="col-6 col-md-4">.col-6 .col-md-4</div>

</div>

<!-- Colunas começam com largura de 50%, em dispositivos móveis, e caem para 33.3% nos desktops -->

<div class="row">

<div class="col-6 col-md-4">.col-6 .col-md-4</div>

<div class="col-6 col-md-4">.col-6 .col-md-4</div>

<div class="col-6 col-md-4">.col-6 .col-md-4</div>

</div>

<!-- Colunas sempre com metade da largura disponível, em dispositivos móveis e desktop -->

<div class="row">

<div class="col-6">.col-6</div>

<div class="col-6">.col-6</div>

</div>

Gutters

Gutters podem ser ajustadas, responsivamente, por utilitários de margem negativa e padding. Para alterar as gutters em uma certa row, combine utilitários de margem negativa (na .row) e um padding correspondente nas .cols.

Aqui está um exemplo de customização do grid Bootstrap no breakpoint large (lg) e acima. Nós aumentamos o padding do .col com .px-lg-5 e, então, o neutralizamos com .mx-lg-n5na .row pai.

Padding de coluna customizado.

Padding de coluna customizado.

Copy

<div class="row mx-lg-n5">

<div class="col py-3 px-lg-5 border bg-light">Padding de coluna customizado.</div>

<div class="col py-3 px-lg-5 border bg-light">Padding de coluna customizado.</div>

</div>

Alinhamento

Use utilitários flexbox de alinhamento para, verticalmente e horizontalmente, alinhar colunas.

Alinhamento vertical

Uma de três colunas

Uma de três colunas

Uma de três colunas

Uma de três colunas

Uma de três colunas

Uma de três colunas

Uma de três colunas

Uma de três colunas

Uma de três colunas

Copy

<div class="container">

<div class="row align-items-start">

<div class="col">

Uma de três colunas

</div>

<div class="col">

Uma de três colunas

</div>

<div class="col">

Uma de três colunas

</div>

</div>

<div class="row align-items-center">

<div class="col">

Uma de três colunas

</div>

<div class="col">

Uma de três colunas

</div>

<div class="col">

Uma de três colunas

</div>

</div>

<div class="row align-items-end">

<div class="col">

Uma de três colunas

</div>

<div class="col">

Uma de três colunas

</div>

<div class="col">

Uma de três colunas

</div>

</div>

</div>

Uma de três colunas

Uma de três colunas

Uma de três colunas

Copy

<div class="container">

<div class="row">

<div class="col align-self-start">

Uma de três colunas

</div>

<div class="col align-self-center">

Uma de três colunas

</div>

<div class="col align-self-end">

Uma de três colunas

</div>

</div>

</div>

Alinhamento horizontal

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Uma de duas colunas

Copy

<div class="container">

<div class="row justify-content-start">

<div class="col-4">

Uma de duas colunas

</div>

<div class="col-4">

Uma de duas colunas

</div>

</div>

<div class="row justify-content-center">

<div class="col-4">

Uma de duas colunas

</div>

<div class="col-4">

Uma de duas colunas

</div>

</div>

<div class="row justify-content-end">

<div class="col-4">

Uma de duas colunas

</div>

<div class="col-4">

Uma de duas colunas

</div>

</div>

<div class="row justify-content-around">

<div class="col-4">

Uma de duas colunas

</div>

<div class="col-4">

Uma de duas colunas

</div>

</div>

<div class="row justify-content-between">

<div class="col-4">

Uma de duas colunas

</div>

<div class="col-4">

Uma de duas colunas

</div>

</div>

</div>

Sem gutters

As gutters, entre colunas, podem ser removidas com uma de nossas classes pré-definidas: .no-gutters. Isto remove a margem negativa do .row e padding horizontal de todas colunas filhas imediatas.

Aqui está o código fonte para criar estes estilos. Perceba que sobrescrição de colunas só são cabíveis na primeira coluna e são selecionadas pelo seletor de atributo. Apesar de gerar um seletor mais específico, padding de coluna ainda pode ser estilizado com utilitários de espaçamento.

**Precisa de um design de ponta-a-ponta?** Deixe as classes .container ou .container-fluid, de lado.

Copy

.no-gutters {

margin-right: 0;

margin-left: 0;

> .col,

> [class\*="col-"] {

padding-right: 0;

padding-left: 0;

}

}

Na prática, é assim que fica. Perceba que você pode continuar a fazer isso com todas as classes grid pré-definidas, incluindo as de largura, breakpoint responsivo, reordenamento e muito mais.

.col-12 .col-sm-6 .col-md-8

.col-6 .col-md-4

Copy

<div class="row no-gutters">

<div class="col-12 col-sm-6 col-md-8">.col-12 .col-sm-6 .col-md-8</div>

<div class="col-6 col-md-4">.col-6 .col-md-4</div>

</div>

Limite de colunas

Se mais de 12 colunas são colocadas dentro de só uma .row, cada grupo de colunas extras vai pular para a próxima linha.

.col-9

.col-4  
Já que 9 + 4 é maior que 12, esta div de quatro colunas pula para essa nova linha.

.col-6  
Colunas subsequentes pulam junto para a nova linha.

Copy

<div class="row">

<div class="col-9">.col-9</div>

<div class="col-4">.col-4<br>Já que 9 + 4 é maior que 12, esta div de quatro colunas pula para essa nova linha.</div>

<div class="col-6">.col-6<br>Colunas subsequentes pulam junto para a nova linha.</div>

</div>

Quebra de colunas

Quebrar colunas, em uma nova linha, exige um pequeno hack: adicionar um elemento com width: 100%, aonde quer que você queira que suas colunas pulem para uma nova linha. Normalmente, isso é conseguido com múltiplos .row, mas isso não se encaixa em toda situação.

.col-6 .col-sm-3

.col-6 .col-sm-3

.col-6 .col-sm-3

.col-6 .col-sm-3

Copy

<div class="row">

<div class="col-6 col-sm-3">.col-6 .col-sm-3</div>

<div class="col-6 col-sm-3">.col-6 .col-sm-3</div>

<!-- Force as próximas colunas quebrarem, em uma nova linha -->

<div class="w-100"></div>

<div class="col-6 col-sm-3">.col-6 .col-sm-3</div>

<div class="col-6 col-sm-3">.col-6 .col-sm-3</div>

</div>

Você também pode aplicar esta quebra em breakpoints específicos, usando nossos utilitários de display responsivos.

.col-6 .col-sm-4

.col-6 .col-sm-4

.col-6 .col-sm-4

.col-6 .col-sm-4

Copy

<div class="row">

<div class="col-6 col-sm-4">.col-6 .col-sm-4</div>

<div class="col-6 col-sm-4">.col-6 .col-sm-4</div>

<!-- Force as próximas colunas quebrarem, em uma nova linha, no breakpoint md ou maior -->

<div class="w-100 d-none d-md-block"></div>

<div class="col-6 col-sm-4">.col-6 .col-sm-4</div>

<div class="col-6 col-sm-4">.col-6 .col-sm-4</div>

</div>

Reordenamento

Classes de ordem

Use classes .order- para controlar a **ordem visual** de seu conteúdo. Estas classes são responsivas, então, você pode definir a ordem por breakpoint (ex: .order-1.order-md-2). Inclui suporte para ordenamento de 1 até 12, em todos breakpoints do grid.

Primeiro, mas não está ordenado.

Era o segundo, mas é o último ordenado.

Era terceiro, mas é o primeiro ordenado.

Copy

<div class="container">

<div class="row">

<div class="col">

Primeiro, mas não está ordenado.

</div>

<div class="col order-12">

Era o segundo, mas é o último ordenado.

</div>

<div class="col order-1">

Era terceiro, mas é o primeiro ordenado.

</div>

</div>

</div>

Também tem as classes responsivas .order-first e .order-last, as quais alteram a order do elemento, aplicando order: -1 e order: 13 (order: $columns + 1), respectivamente. Estas classes também podem ser misturadas com as classes numeradas .order-\*, se necessário.

Era primeiro, mas ordenado como último.

Segundo, mas sem ordenamento.

Era terceiro, mas ordenado como primeiro.

Copy

<div class="container">

<div class="row">

<div class="col order-last">

Era primeiro, mas ordenado como último.

</div>

<div class="col">

Segundo, mas sem ordenamento.

</div>

<div class="col order-first">

Era terceiro, mas ordenado como primeiro.

</div>

</div>

</div>

Deslocando colunas

Você pode deslocar colunas, de duas maneiras: com nossas classes responsivas .offset- e nossos utilitários de margem. Classes do grid podem ser combinadas para dimensionar colunas, enquato classes de margem são mais úteis para rápidos layouts, onde a largura do deslocamento é variável.

Classes offset

Mova colunas para a direita, usando classes .offset-md-\*. Estas classes crescem a margem esquerda da coluna, em \* colunas. Por exemplo , .offset-md-4 desloca o elemento .col-md-4, por quatro colunas.

.col-md-4

.col-md-4 .offset-md-4

.col-md-3 .offset-md-3

.col-md-3 .offset-md-3

.col-md-6 .offset-md-3

Copy

<div class="row">

<div class="col-md-4">.col-md-4</div>

<div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>

</div>

<div class="row">

<div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>

<div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>

</div>

<div class="row">

<div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>

</div>

Além da limpeza de coluna, em breakpoints responsivos, você pode precisar resetar deslocamentos. Veja isso, em ação, no grid exemplo.

.col-sm-5 .col-md-6

.col-sm-5 .offset-sm-2 .col-md-6 .offset-md-0

.col-sm-6 .col-md-5 .col-lg-6

.col-sm-6 .col-md-5 .offset-md-2 .col-lg-6 .offset-lg-0

Copy

<div class="row">

<div class="col-sm-5 col-md-6">.col-sm-5 .col-md-6</div>

<div class="col-sm-5 offset-sm-2 col-md-6 offset-md-0">.col-sm-5 .offset-sm-2 .col-md-6 .offset-md-0</div>

</div>

<div class="row">

<div class="col-sm-6 col-md-5 col-lg-6">.col-sm-6 .col-md-5 .col-lg-6</div>

<div class="col-sm-6 col-md-5 offset-md-2 col-lg-6 offset-lg-0">.col-sm-6 .col-md-5 .offset-md-2 .col-lg-6 .offset-lg-0</div>

</div>

Utilitários de margem

Graças a migração para o flexbox, na v4, você você usar utilitários de margem (como .mr-auto) para forçar colunas irmãs se distanciarem, uma das outras.

.col-md-4

.col-md-4 .ml-auto

.col-md-3 .ml-md-auto

.col-md-3 .ml-md-auto

.col-auto .mr-auto

.col-auto

Copy

<div class="row">

<div class="col-md-4">.col-md-4</div>

<div class="col-md-4 ml-auto">.col-md-4 .ml-auto</div>

</div>

<div class="row">

<div class="col-md-3 ml-md-auto">.col-md-3 .ml-md-auto</div>

<div class="col-md-3 ml-md-auto">.col-md-3 .ml-md-auto</div>

</div>

<div class="row">

<div class="col-auto mr-auto">.col-auto .mr-auto</div>

<div class="col-auto">.col-auto</div>

</div>

Aninhamento

Para aninhar seu conteúdo no grid padrão, coloque um .row com um conjunto de colunas .col-sm-\*, dentro de uma coluna .col-sm-\* existente. Rows aninhados devem ter um conjunto de 12 colunas ou menos (não é obrigatório usar todo o espaço disponível para 12 colunas).

Level 1: .col-sm-9

Level 2: .col-8 .col-sm-6

Level 2: .col-4 .col-sm-6

Copy

<div class="row">

<div class="col-sm-9">

Level 1: .col-sm-9

<div class="row">

<div class="col-8 col-sm-6">

Level 2: .col-8 .col-sm-6

</div>

<div class="col-4 col-sm-6">

Level 2: .col-4 .col-sm-6

</div>

</div>

</div>

</div>

Mixins Sass

Quando usando os arquivos fonte Sass do Bootstrap, você tem a opção de usar variáveis e mixins Sass para criar layouts personalizados, semânticos e responsivos. Nossas classes grid pré-definidas usam as mesmas variáveis e mixins para gerarem todo um conjunto de classes prontas para usar, em layouts responsivos rápidos.

Variáveis

Variáveis e mapas determinam o número de colunas, largura da gutter e o ponto media query de começar a flutuar colunas. Nós usamos elas para gerar as classes pré-definidas mostradas acima, assim como os mixins listados abaixo.

Copy

$grid-columns: 12;

$grid-gutter-width: 30px;

$grid-breakpoints: (

// Tela extra small (muito pequena, no português)

xs: 0,

// Tela small (pequena, no português)

sm: 576px,

// Tela média

md: 768px,

// Tela large (grande, no português)

lg: 992px,

// Tela extra large (muito grande, no português)

xl: 1200px

);

$container-max-widths: (

sm: 540px,

md: 720px,

lg: 960px,

xl: 1140px

);

Mixins

Mixins são usados em conjunto com as variáveis grid para gerar CSS semântico, em colunas individuais.

Copy

// Criar um elemento pai para conjuntos de colunas.

@include make-row();

// Apronta o elemnto grid (aplica tudo, menos largura).

@include make-col-ready();

@include make-col($size, $columns: $grid-columns);

// Seja chique, usando deslocamento ou ordenamento Bootstrap.

@include make-col-offset($size, $columns: $grid-columns);

Exemplo de uso

Você pode modificar as variáveis para seus próprios valores ou, apenas, usar os mixins com seus valores padrões. Aqui está um exemplo, usando configurações padrões para criar um layout de duas colunas e um espaço, entre elas.

Copy

.container-exemplo {

width: 800px;

@include make-container();

}

.row-exemplo {

@include make-row();

}

.exemplo-conteudo-principal {

@include make-col-ready();

@include media-breakpoint-up(sm) {

@include make-col(6);

}

@include media-breakpoint-up(lg) {

@include make-col(8);

}

}

.exemplo-conteudo-secundario {

@include make-col-ready();

@include media-breakpoint-up(sm) {

@include make-col(6);

}

@include media-breakpoint-up(lg) {

@include make-col(4);

}

}

Conteúdo principal

Conteúdo secundário

Copy

<div class="container-exemplo">

<div class="example-row">

<div class="example-content-main">Conteúdo principal</div>

<div class="example-content-secondary">Conteúdo secundário</div>

</div>

</div>

Personalizando o grid

Usando nossas variáveis e mapas Sass integrados, é possível customizar as classes grid pré-definidas, completamente. Altere a quantia de breakpoints, dimensões de media queries, largura de containers e recompile.

Colunas e gutters

O número de colunas pode ser modificado com variáveis Sass. $grid-columns é usada para gerar as larguras (em porcentagem) de cada coluna individual, enquanto $grid-gutter-widthgera larguras para a coluna gutters.

Copy

$grid-columns: 12 !default;

$grid-gutter-width: 30px !default;

Breakpoints

Além das colunas, você pode querer customizar o número de breakpoints. Se quiser só quatro deles, você atualizar $grid-breakpoints e $container-max-widths para algo assim:

Copy

$grid-breakpoints: (

xs: 0,

sm: 480px,

md: 768px,

lg: 1024px

);

$container-max-widths: (

sm: 420px,

md: 720px,

lg: 960px

);

Quando fazendo mudanças nos mapas ou variáveis Sass, você vai precisar salvar as mudanças e recompilar. Fazendo isso, um novo conjunto de classes pré-definidas para larguras, deslocamento e ordenamento, será gerado. Utilitários de visibilidade responsivos também serão atualizados para usarem os breakpoints customizados. Se assegure de colocar os valores grid em px (não em rem, em ou %).