



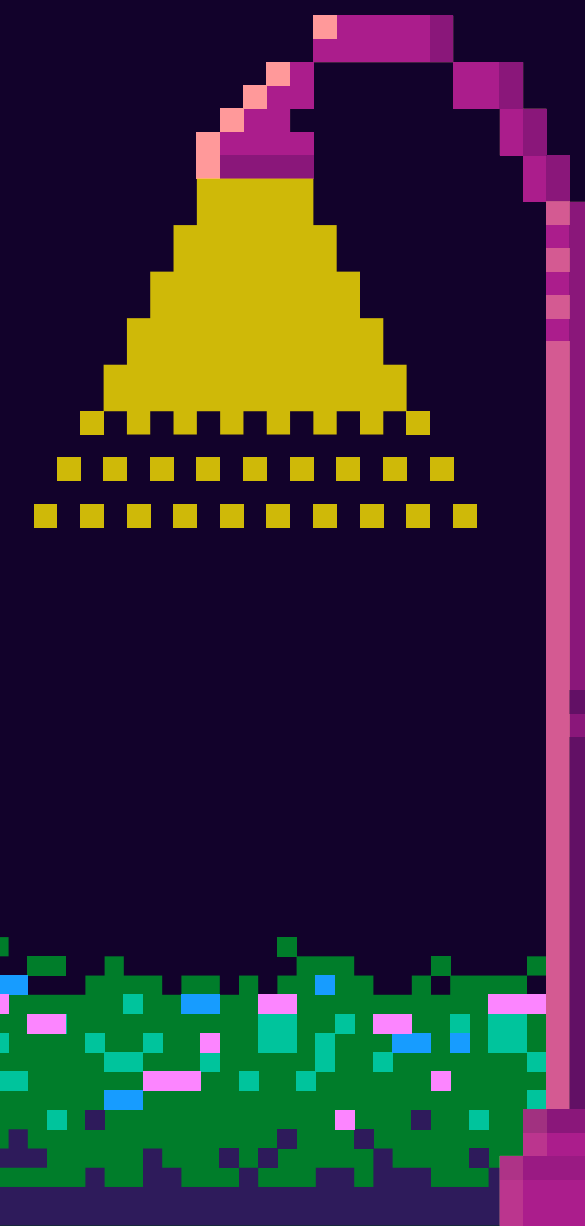
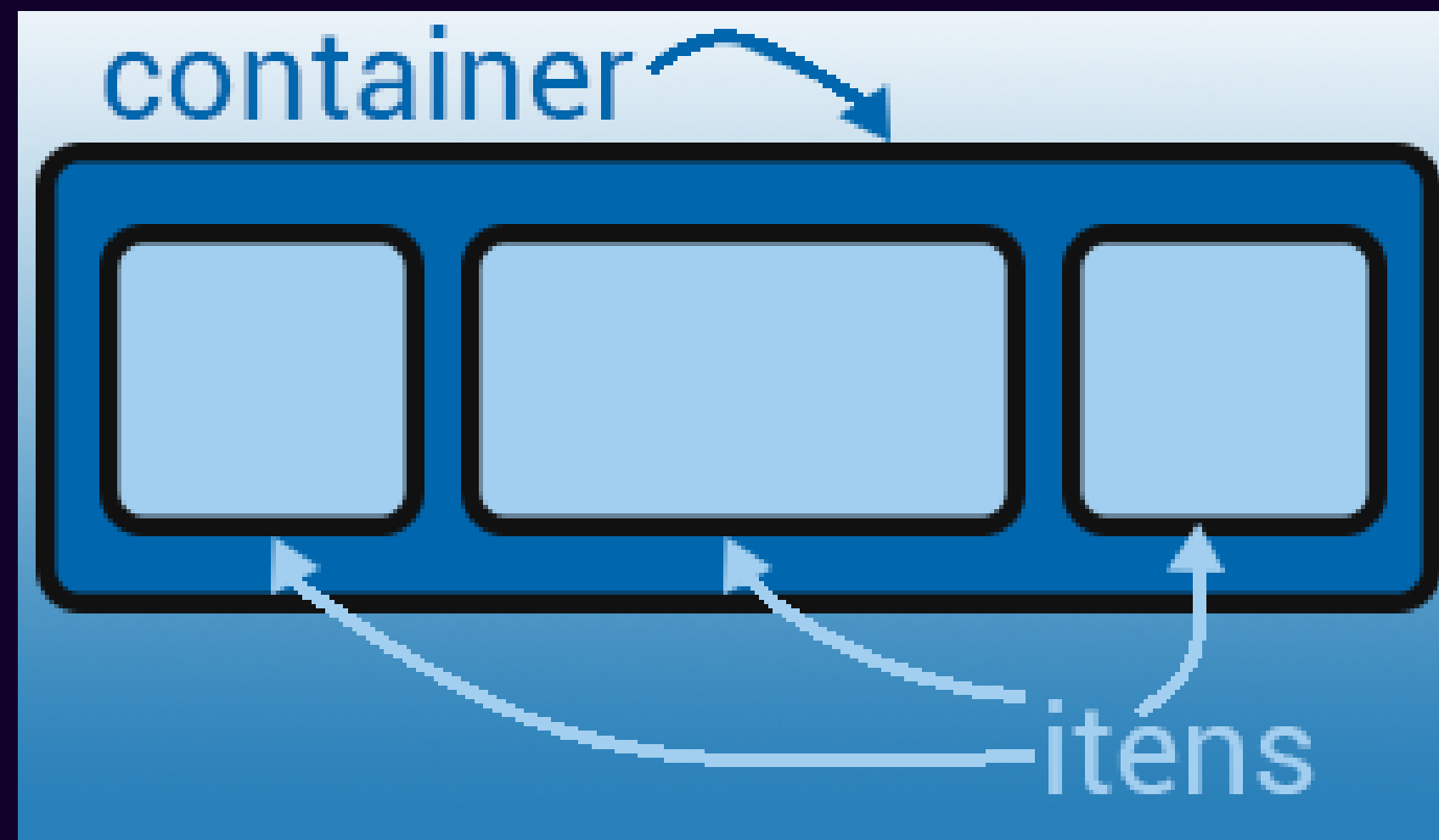
HELLO WORLD



FLEXBOOK

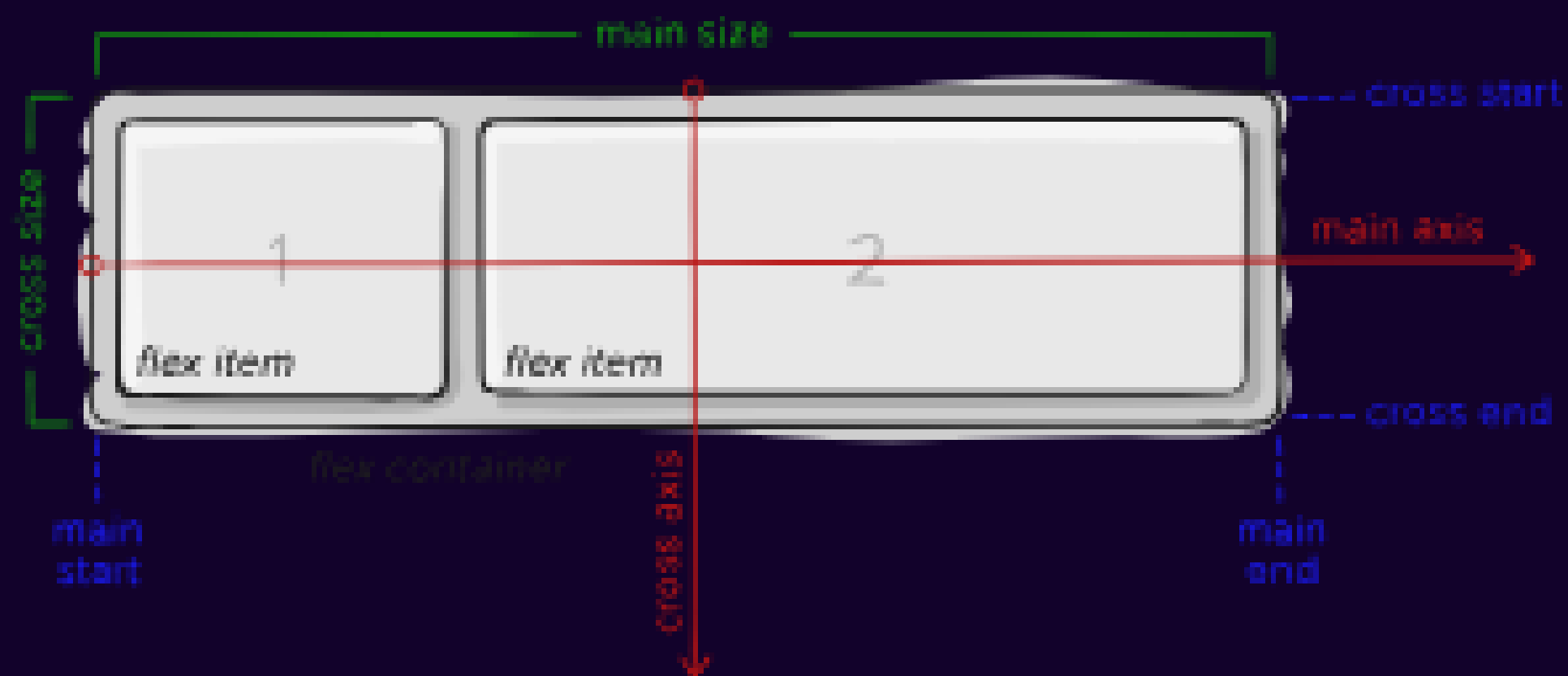
O QUE É FLEXBOX?

Flexbox é um modelo de layout do CSS que facilita a criação de layouts flexíveis e responsivos. Ele permite organizar elementos dentro de um container de forma eficiente, distribuindo espaço e alinhando os itens automaticamente, independentemente do tamanho da tela.



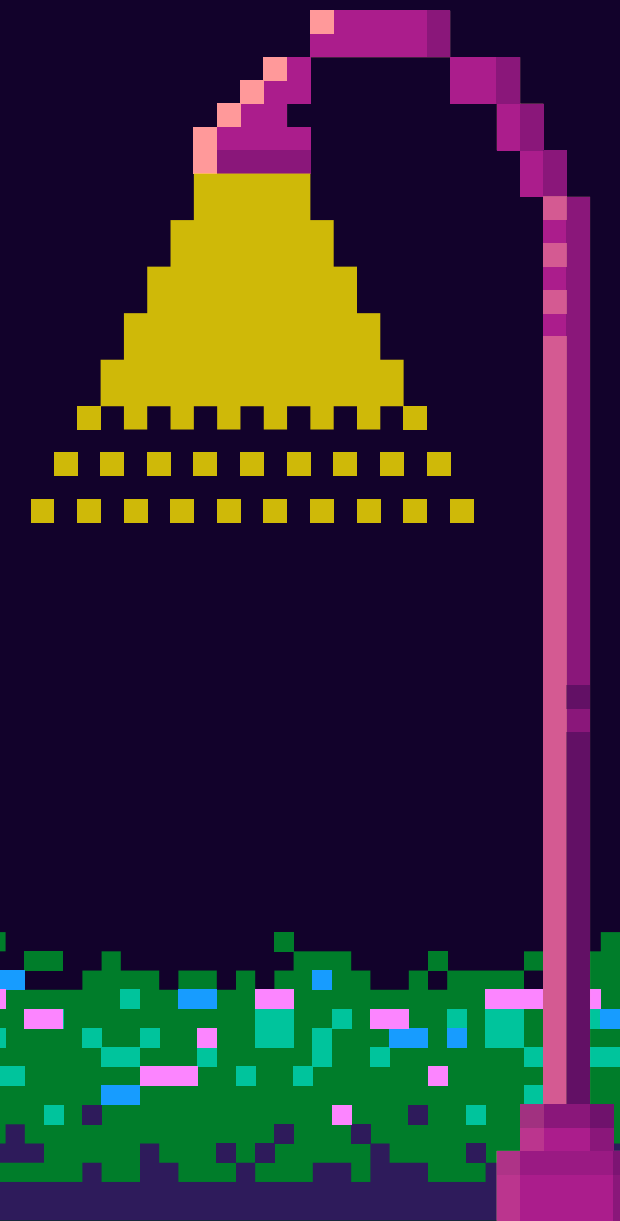
ELEMENTOS

O Flexbox é um módulo completo e não uma única propriedade; algumas delas devem ser declaradas no container (o elemento-pai, que chamamos de flex container), enquanto outras devem ser declaradas nos elementos-filhos (os flex itens). Se o leiaute "padrão" é baseado nas direções block e inline, o leiaute Flex é baseado em direções "flex flow". Veja abaixo um diagrama da especificação, explicando a ideia central por trás do leiaute Flex.



Os ítems serão dispostos no leiaute seguindo ou o eixo principal ou o transversal.

- Eixo principal: o eixo principal de um flex container é o eixo primário e ao longo dele são inseridos os flex items. Cuidado: O eixo principal não é necessariamente horizontal; vai depender da propriedade flex-direction (veja abaixo).
- main-start | main-end: os flex items são inseridos dentro do container começando pelo lado start, indo em direção ao lado end.
- Tamanho principal: A largura ou altura de um flex item, dependendo da direção do container, é o tamanho principal do ítem. A propriedade de tamanho principal de um flex item pode ser tanto width quanto height, dependendo de qual delas estiver na direção principal.
- Eixo transversal: O eixo perpendicular ao eixo principal é chamado de eixo transversal. Sua direção depende da direção do eixo principal.
- cross-start | cross-end: Linhas flex são preenchidas com ítems e adicionadas ao container, começando pelo lado cross start do flex container em direção ao lado cross end.
- cross size: A largura ou altura de um flex item, dependendo do que estiver na dimensão transversal, é o cross size do ítem. A propriedade cross size pode ser tanto a largura quanto a altura do ítem, o que estiver na transversal.



Flex container

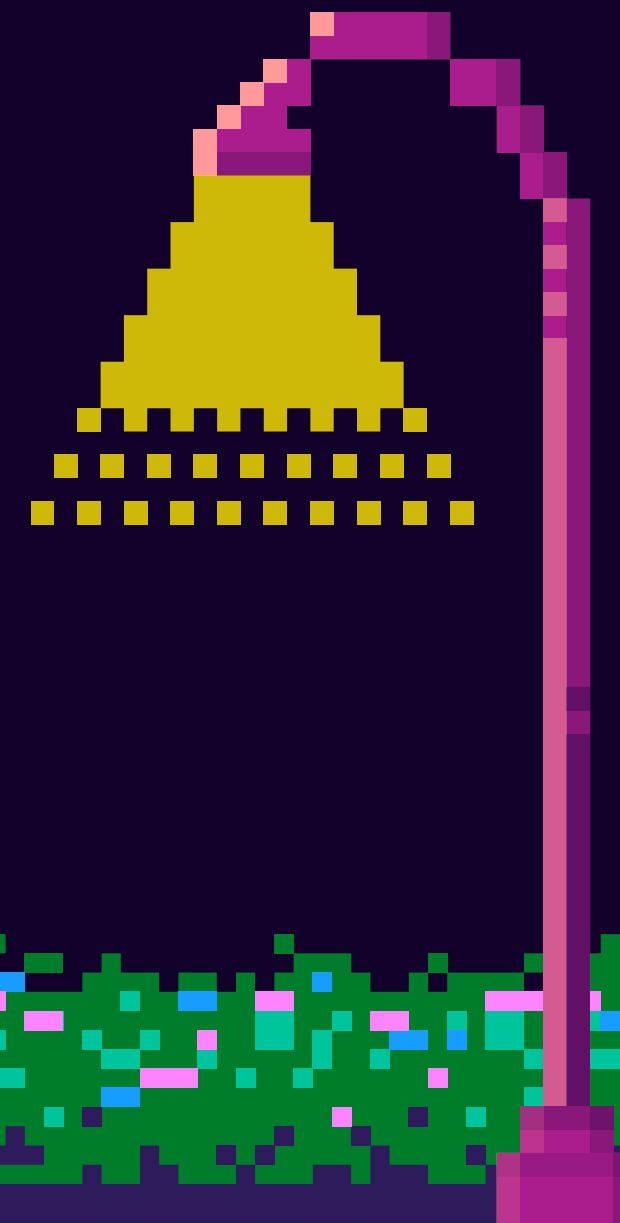
Flex container é o elemento que envolve sua estrutura. Você define que um elemento é um Flex Container com a propriedade `display` e valores `flex` ou `inline-flex`.

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

```
.flex-container {
  display: flex;
}
```

Flex Item são elementos-filhos do flex container.

Eixos ou Axes são as duas direções básicas que existem em um Flex Container: main axis, ou eixo principal, e cross axis, ou eixo transversal.



PROPRIEDADES PARA O ELEMENTO-PAI

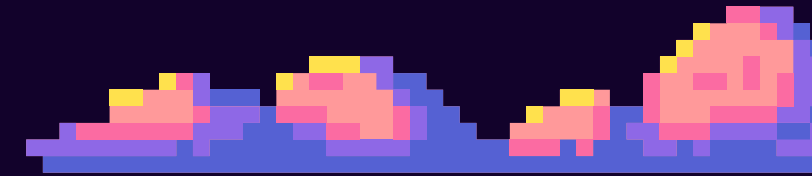
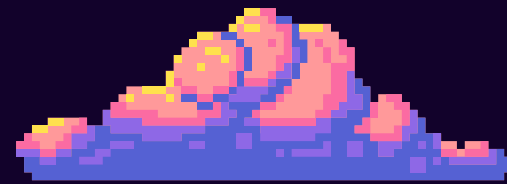
Quando utilizamos o Flexbox, é muito importante saber quais propriedades são declaradas no elemento-pai (por exemplo, uma div que irá conter os elementos a serem alinhados) e quais serão declaradas nos elementos-filhos. Abaixo, seguem propriedades que devem ser declaradas utilizando o elemento-pai como seletor (para alinhar elementos-filhos).



display

Esta propriedade define um flex container; inline ou block dependendo dos valores passados. Coloca todos os elementos-filhos diretos num contexto Flex.

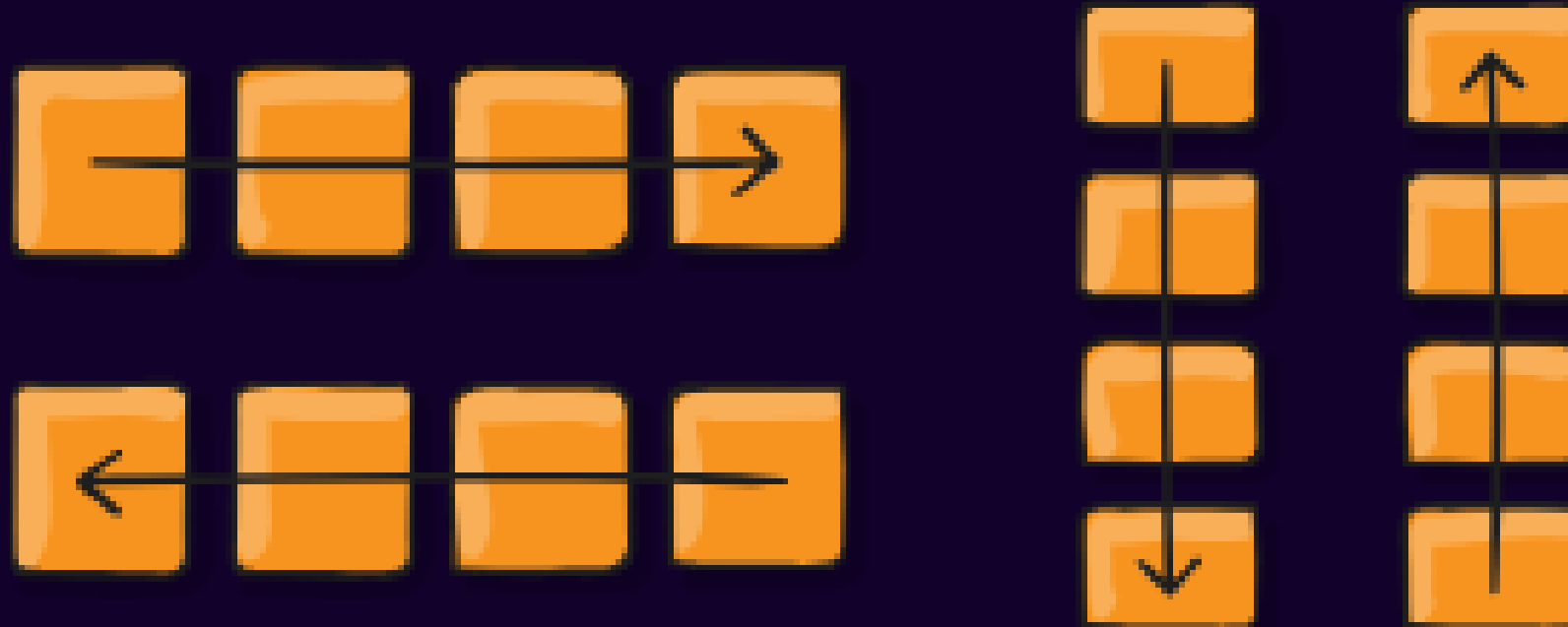
```
.container {  
  display: flex; /* or inline-flex */  
}
```



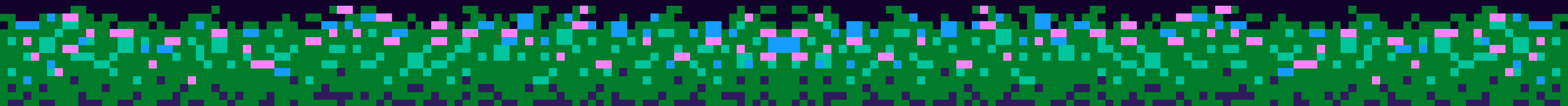
FLEX-DIRECTION

Estabelece o eixo principal, definindo assim a direção em que os flex items são alinhados no flex container. O Flexbox é (com exceção de um wrapping opcional) um conceito de layout de uma só direção. Pense nos flex items inicialmente posicionais ou em linhas horizontais ou em colunas verticais.

```
.flex-container {  
  flex-direction: row | row-reverse | column | column-reverse;  
}
```



- row (padrão): esquerda para a direita em ltr (left to right), direita para a esquerda em rtl (right to left)
- row-reverse: direita para a esquerda em ltr, esquerda para a direita em rtl
- column: mesmo que row, mas de cima para baixo
- column-reverse: mesmo que row-reverse mas de baixo para cima

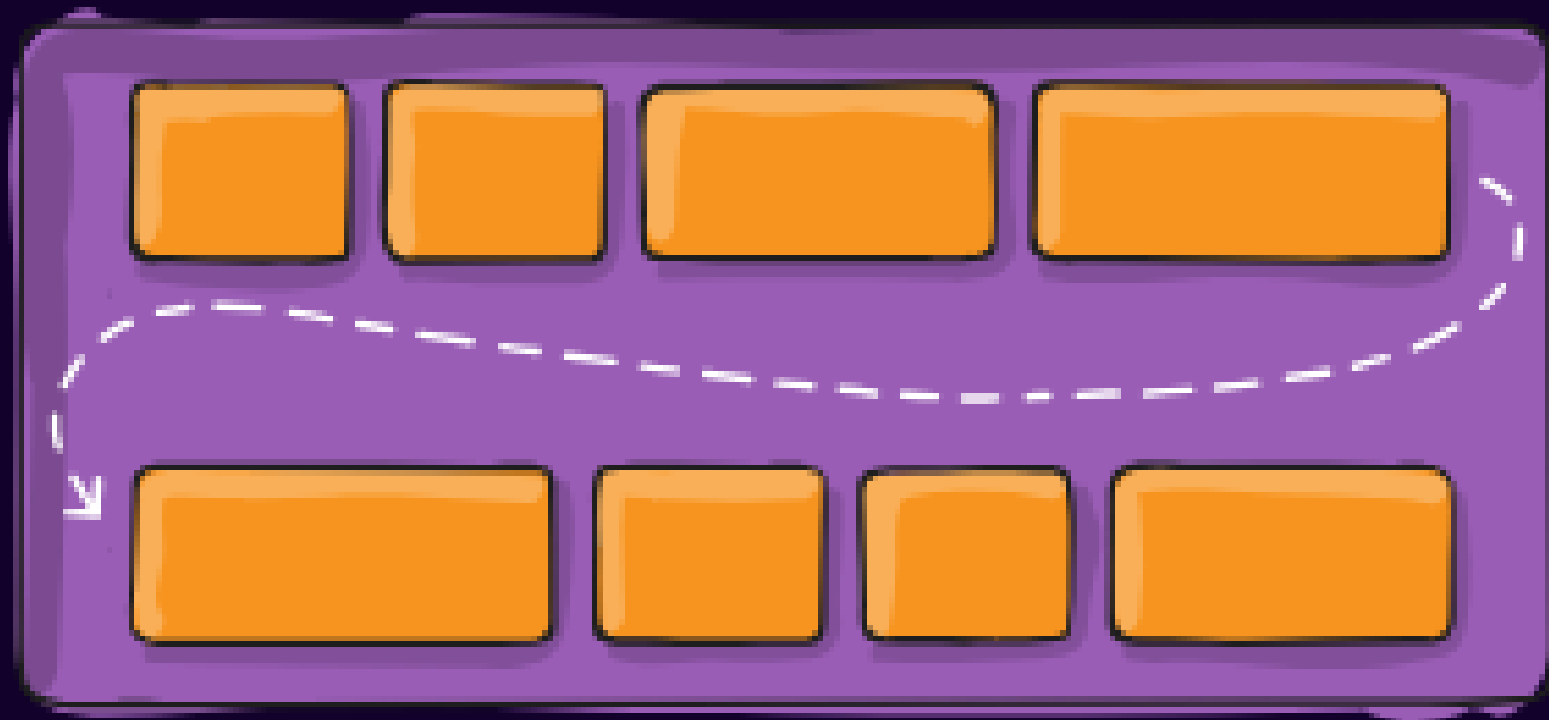


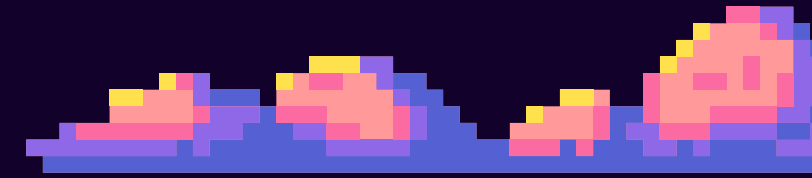
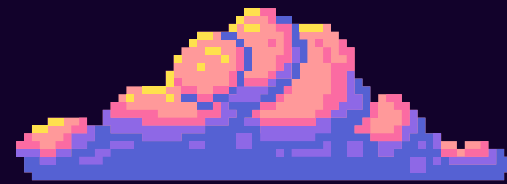
FLEX-WRAP

Por padrão, os flex items vão todos tentar se encaixar em uma só linha. Com esta propriedade você pode modificar esse comportamento e permitir que os itens quebrem para uma linha seguinte conforme for necessário.

```
.flex-container {  
  flex-wrap: nowrap | wrap | wrap-reverse;  
}
```

- nowrap (padrão): todos os flex items ficarão em uma só linha
- wrap: os flex items vão quebrar em múltiplas linhas, de cima para baixo
- wrap-reverse: os flex items vão quebrar em múltiplas linhas de baixo para cima

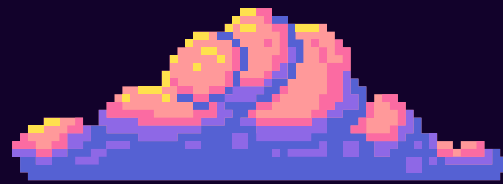




FLEX-FLOW

A propriedade flex-flow é uma propriedade shorthand (uma mesma declaração inclui vários valores relacionados a mais de uma propriedade) que inclui flex-direction e flex-wrap. Determina quais serão os eixos principal e transversal do container. O valor padrão é row nowrap.

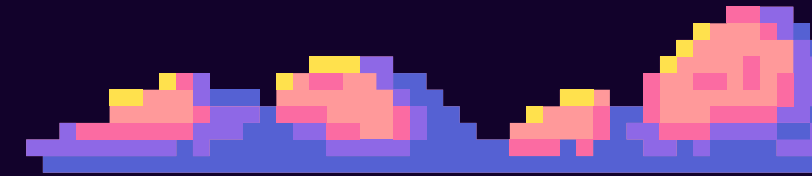
```
.flex-container {  
  flex-flow: row nowrap | row wrap | column nowrap | column wrap;  
}
```



JUSTIFY-CONTENT

Esta propriedade define o alinhamento dos ítems ao longo do eixo principal. Ajuda a distribuir o espaço livre que sobrar no container tanto se todos os flex items em uma linha são inflexíveis, ou são flexíveis mas já atingiram seu tamanho máximo. Também exerce algum controle sobre o alinhamento de ítems quando eles ultrapassam o limite da linha.

- flex-start (padrão): os ítems são alinhados junto à borda de início (start) de acordo com qual for a flex-direction do container.
- flex-end: os ítems são alinhados junto à borda final (end) de acordo com qual for a flex-direction do container.
- start: os ítems são alinhados junto à borda de início da direção do writing-mode (modo de escrita).
- end: os ítems são alinhados junto à borda final da direção do writing-mode (modo de escrita).
- left: os ítems são alinhados junto à borda esquerda do container, a não ser que isso não faça sentido com o flex-direction que estiver sendo utilizado. Nesse caso, se comporta como start.
- right: os ítems são alinhados junto à borda direita do container, a não ser que isso não faça sentido com o flex-direction que estiver sendo utilizado. Nesse caso, se comporta como start.
- center: os ítems são centralizados na linha.
- space-between: os ítems são distribuídos de forma igual ao longo da linha; o primeiro ítem junto à borda inicial da linha, o último junto à borda final da linha.
- space-around: os ítems são distribuídos na linha com o mesmo espaçamento entre eles. Note que, visualmente, o espaço pode não ser igual, uma vez que todos os ítems tem a mesma quantidade de espaço dos dois lados: o primeiro item vai ter somente uma unidade de espaço junto à borda do container, mas duas unidades de espaço entre ele e o próximo ítem, pois o próximo ítem também tem seu próprio espaçamento que está sendo aplicado.
- space-evenly: os ítems são distribuídos de forma que o espaçamento entre quaisquer dois itens da linha (incluindo entre os ítems e as bordas) seja igual.



flex-start



flex-end



center



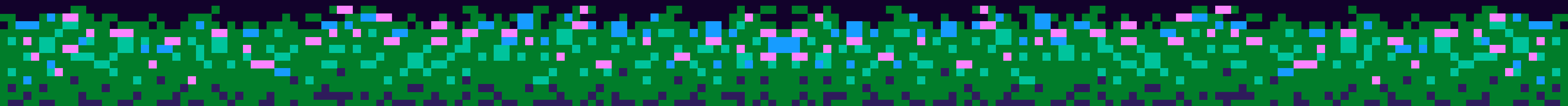
space-between

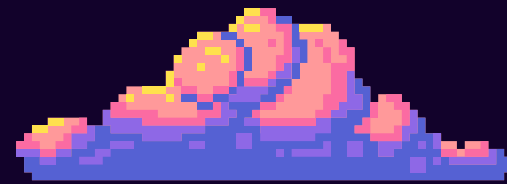


space-around



space-evenly

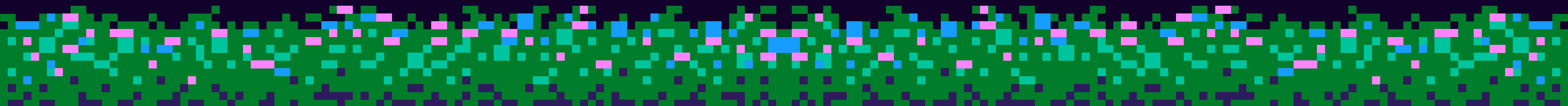
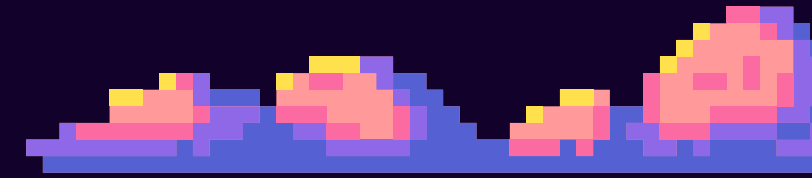
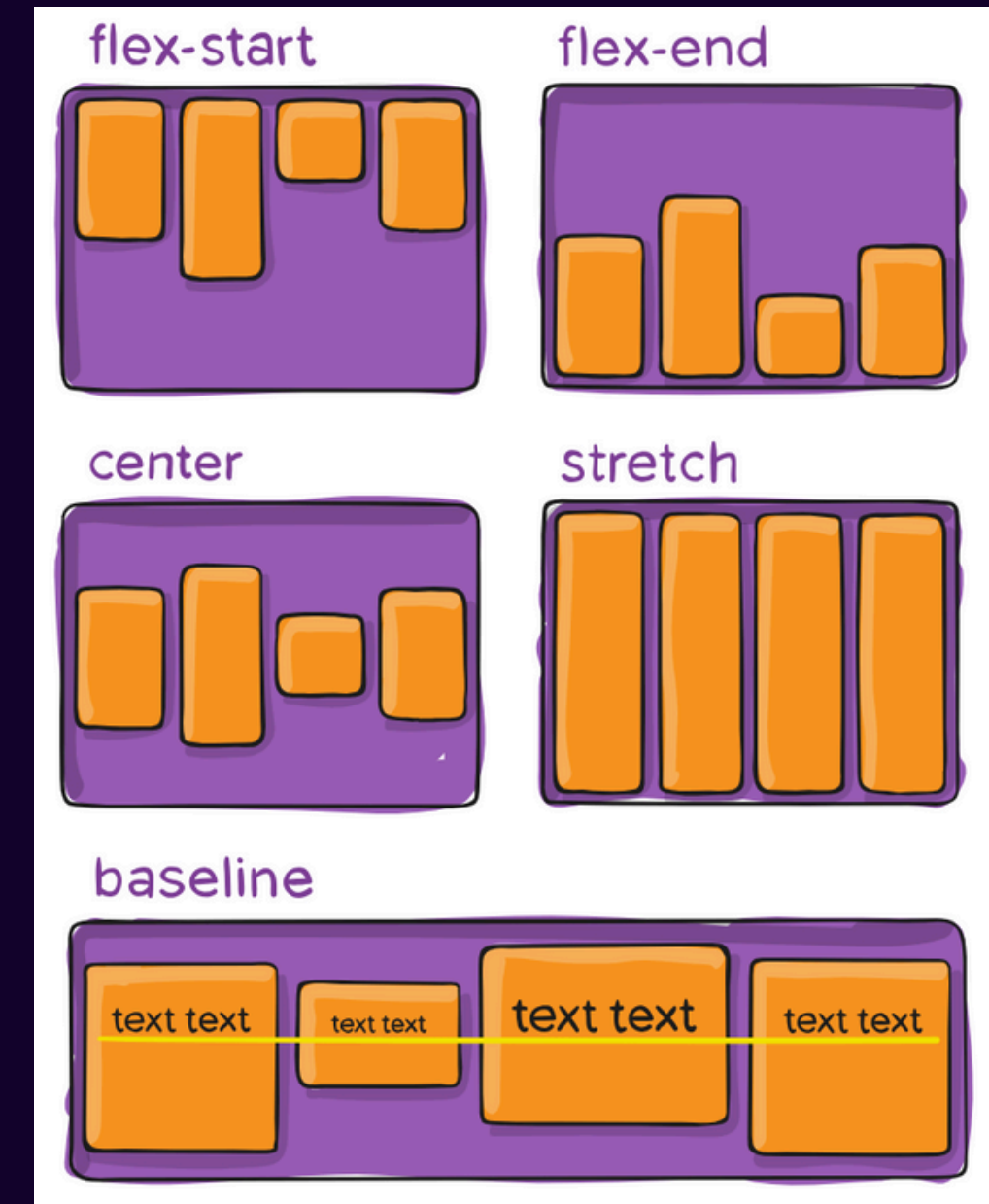


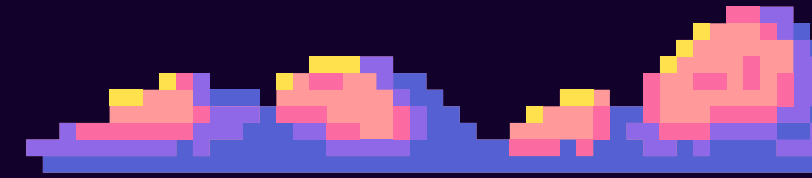
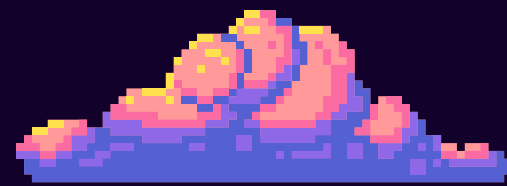


ALIGN-ITEMS

define o comportamento padrão de como flex items são alinhados de acordo com o eixo transversal (cross axis). De certa forma, funciona de forma similar ao justify-content, porém no eixo transversal (perpendicular ao eixo principal).

- stretch (padrão): estica os itens para preencher o container, respeitando o min-width/max-width).
- flex-start/ start / self-start: itens são posicionados no início do eixo transversal. A diferença entre eles é sutil e diz respeito às regras de flex-direction ou writing-mode.
- center: itens são centralizados no eixo transversal.
- baseline: itens são alinhados de acordo com suas baselines.



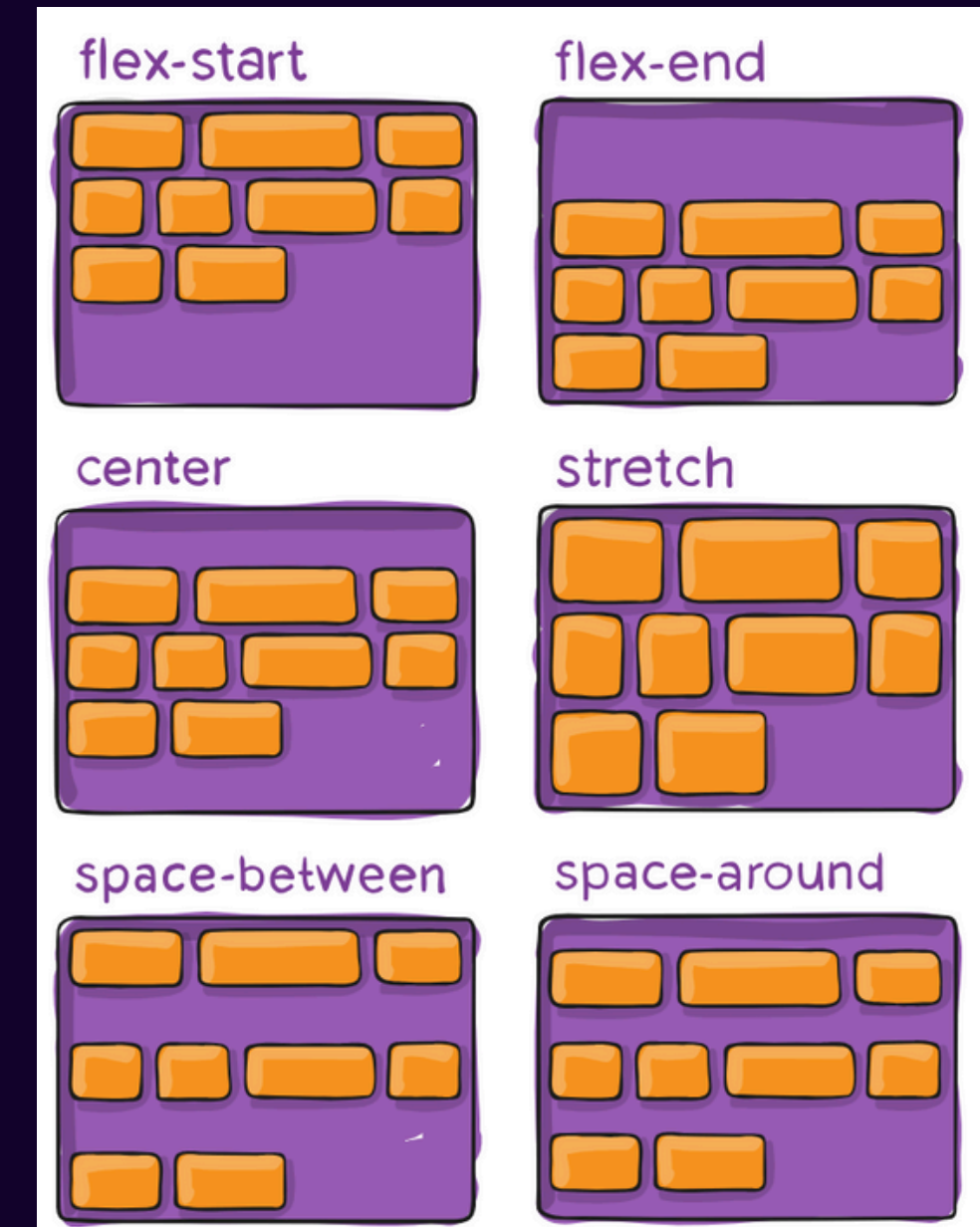


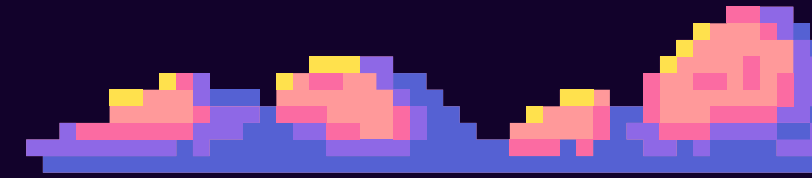
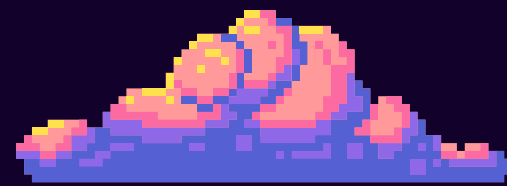
ALIGN-CONTENT

Organiza as linhas dentro de um flex container quando há espaço extra no eixo transversal, similar ao modo como justify-content alinha ítems individuais dentro do eixo principal.

Importante: Esta propriedade não tem efeito quando há somente uma linha de flex items no container.

- flex-start / start: ítems alinhados com o início do container. O valor (com maior suporte dos navegadores) flex-start se guia pela flex-direction, enquanto start se guia pela direção do writing-mode.
- flex-end / end: ítems alinhados com o final do container. O valor (com maior suporte dos navegadores) flex-end se guia pela flex-direction, enquanto end se guia pela direção do writing-mode.
- center: ítems centralizados no container.
- space-between: ítems distribuídos igualmente; a primeira linha junto ao início do container e a última linha junto ao final do container.
- space-around: ítems distribuídos igualmente com o mesmo espaçamento entre cada linha.
- space-evenly: ítems distribuídos igualmente com o mesmo espaçamento entre eles.
- stretch (padrão): ítems em cada linha esticam para ocupar o espaço remanescente entre elas.





PROPRIEDADES PARA ELEMENTOS-FILHOS

A seguir, veremos propriedades que devem ser declaradas tendo como seletor os elementos-filhos, ou seja:

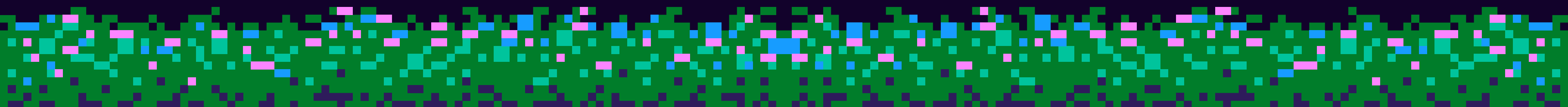
```
<div class="flex-container">  
  <div class="flex-item">1</div>  
  <div class="flex-item">2</div>  
  <div class="flex-item">3</div>  
</div>
```

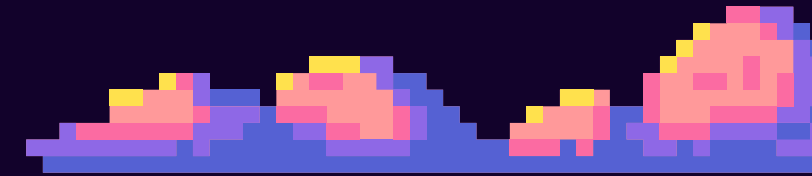
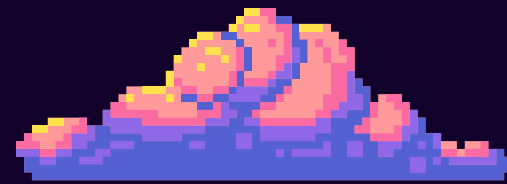
Isso significa que, onde existe um elemento-pai com propriedade flex (o flex-container), é possível atribuir propriedades flex específicas também para as elementos-filhos (flex-item). Você pode definir as propriedades abaixo para apenas um dos elementos-filhos através de um identificador, como uma classe específica.

order



```
.flex-item {  
  order: <número>; /* o valor padrão é 0 */  
}
```

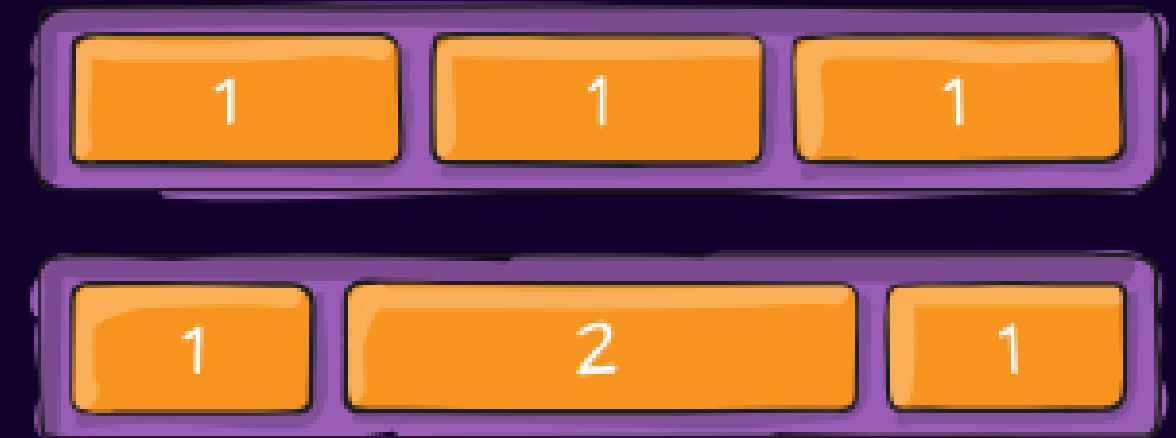




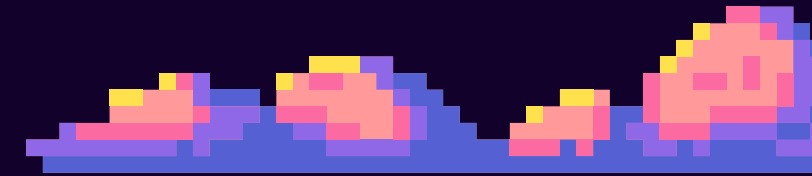
FLEX-GROW

Define a habilidade de um flex item de crescer, caso necessário. O valor dessa propriedade é um valor numérico sem indicação de unidade, que serve para cálculo de proporção. Este valor dita a quantidade de espaço disponível no container que será ocupado pelo item.

Se todos os itens tiverem flex-grow definido em 1, o espaço remanescente no container será distribuído de forma igual entre todos. Se um dos itens tem o valor de 2, vai ocupar o dobro de espaço no container com relação aos outros (ou pelo menos vai tentar fazer isso).



```
.flex-item {  
  flex-grow: <numero>; /* o valor default(padrão) é 0 */  
}
```

FLEX-SHRINK

Define a habilidade de um flex item de encolher, caso necessário.

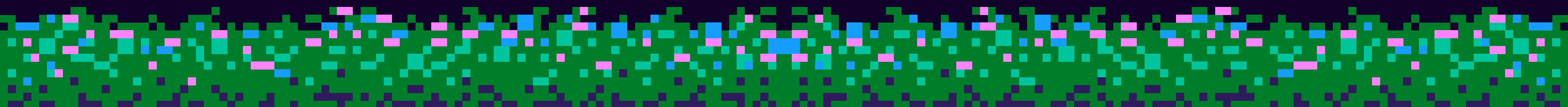
```
.flex-item {  
  flex-shrink: <número>; /* o valor padrão é 0 */  
}
```

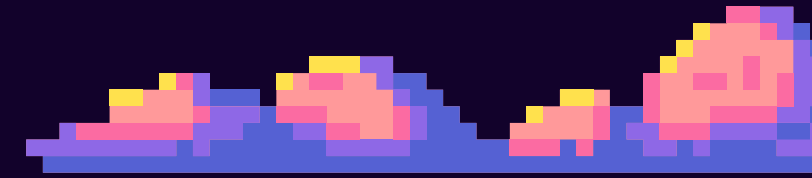
flex-basis

Define o tamanho padrão para um elemento antes que o espaço remanescente do container seja distribuído. Pode ser um comprimento (por exemplo, 20%, 5rem, etc) ou uma palavra-chave. A palavra-chave auto significa "observe minhas propriedades de altura ou largura". A palavra-chave content significa "estabeleça o tamanho com base no conteúdo interno do item".

```
.flex-item {  
  flex-basis: flex-basis: | auto;  
}
```

o valor padrão é auto





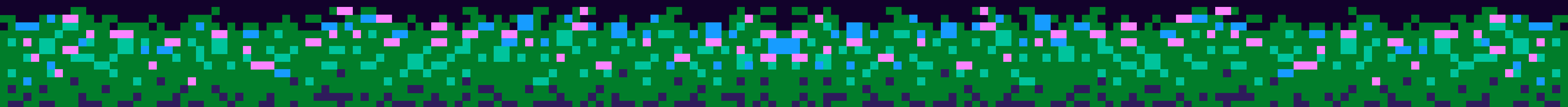
FLEX

Esta é a propriedade shorthand para flex-grow, flex-shrink e flex-basis, combinadas. O segundo e terceiro parâmetros (flex-shrink e flex-basis) são opcionais. O padrão é 0 1 auto, mas se você definir com apenas um número, é equivalente a 0 1.

```
.item {  
  flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ]  
}
```

É recomendado que você utilize esta propriedade shorthand ao invés de definir cada uma das propriedades em separado. O shorthand define os outros valores de forma inteligente.

obs: shorthand significa abreviação, é um termo muito usado na programação

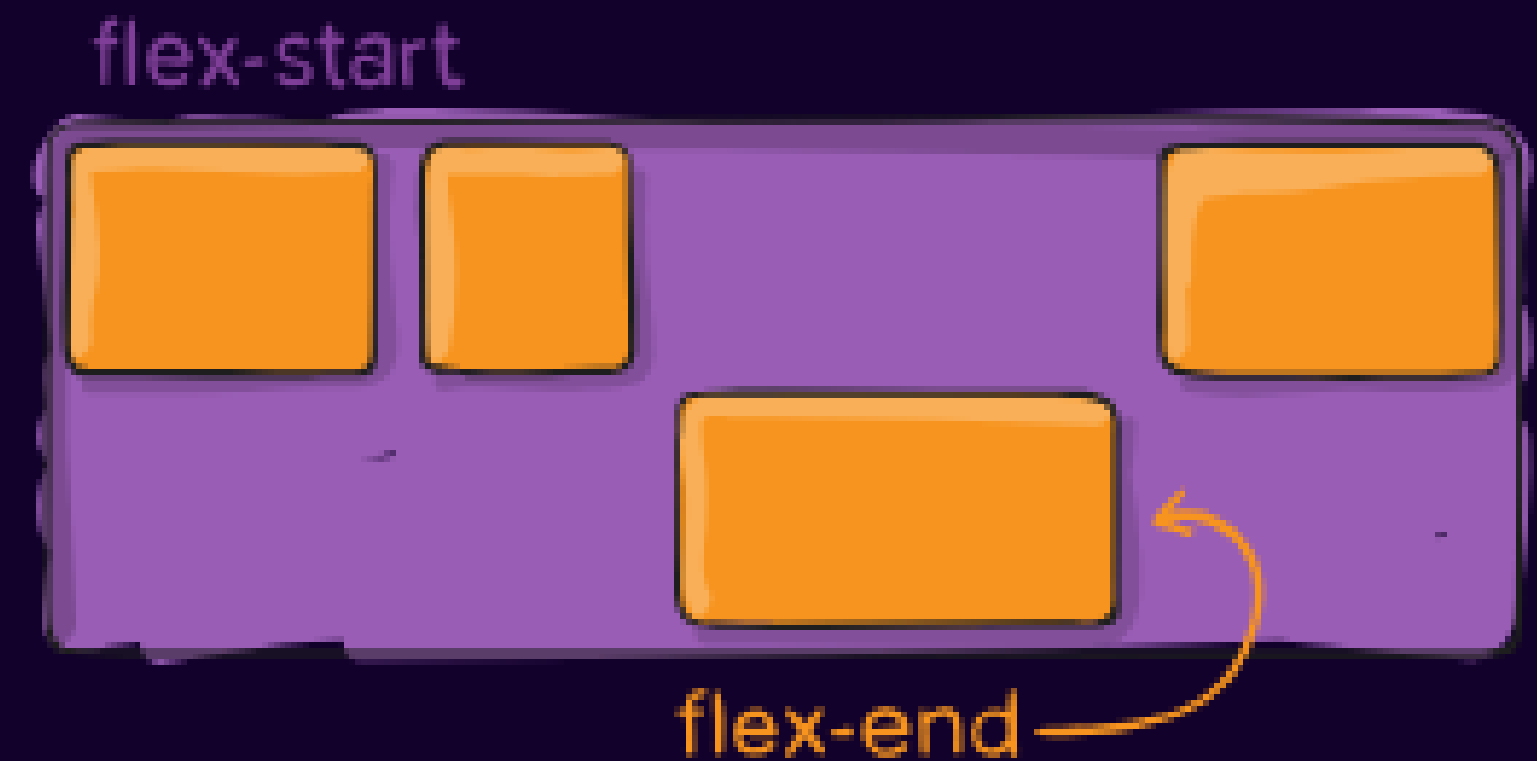


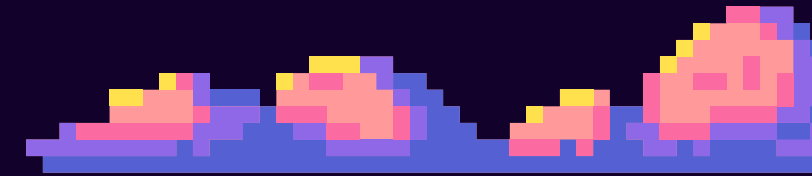
ALIGN-SELF

Permite que o alinhamento padrão (ou o que estiver definido por `align-items`) seja sobrescrito para itens individuais.

Por favor veja a explicação da propriedade `align-items` para entender quais são os possíveis valores.

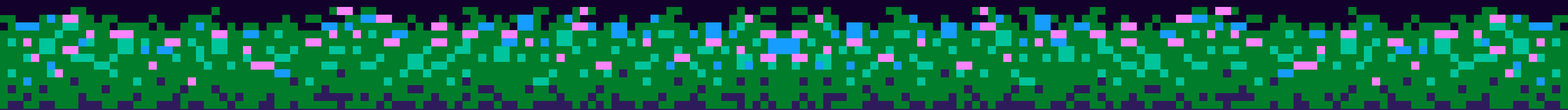
```
.item {  
  align-self: auto | flex-start | flex-end | center | baseline | stretch;  
}
```





IMPORTANTE!

- O CSS só enxerga a hierarquia pai-filho; não vai aplicar as propriedades Flex para elementos que não estejam diretamente relacionados;
- Para que as propriedades funcionem nos elementos-filhos, as pais devem ter propriedade display: flex;
- As propriedades float, clear e vertical-align não têm efeito em flex-items.





EODAA

LOGAA

Flexbox Froggy.

EXERCÍCIO


Get **insights** that help
your business grow.

Discover the benefits of data analytics and make
better decisions regarding revenue, customer
experience, and overall efficiency.

10k+
COMPANIES

314
TEMPLATES

12M+
QUERIES



[LINK PARA O DESAFIO](#)

NOSSOS CONTATOS:



27 99500-7495



<https://beacons.ai/prismatech>



producaoprismatech@gmail.com



Avenida Jerônimo Monteiro 145, Vitória

