Out-of-flow Positioned Fragmentation

Alison Maher almaher@microsoft.com

Fragmentation Overview

- Fragmentainer
 - Fragment container
 - A box that contains a portion of fragmented content

- Fragmentainer
 - Fragment container
 - A box that contains a portion of fragmented content
- Fragmentation contexts
 - Collection of fragmentainers

- Fragmentainer
 - Fragment container
 - A box that contains a portion of fragmented content
- Fragmentation contexts
 - Collection of fragmentainers
- Fragmentation
 - "The process of splitting a content flow across the <u>fragmentainers</u> that form a <u>fragmentation</u> <u>context</u>."

Paged Media

Fragmentainer = page

Fragmentation context =Collection of pages

8/7/20

CSS Fragmentation Module Level 3

CSS Fragmentation Module Level 3

W3C Candidate Recommendation, 4 December 2018



This version:

https://www.w3.org/TR/2018/CR-css-break-3-20181204/

Latest published version:

https://www.w3.org/TR/css-break-3/

Editor's Draft:

https://drafts.csswg.org/css-break/

Previous Versions:

https://www.w3.org/TR/2017/CR-css-break-3-20170209/ https://www.w3.org/TR/2016/CR-css-break-3-20160114/

Test Suite:

http://test.csswg.org/suites/css-break-3 dev/nightly-unstable/

Editors

Rossen Atanassov (Microsoft)
Elika J. Etemad / fantasai (Invited Expert)

Issue Tracking:

GitHub Issue:

Copyright © 2018 W3C[®] (MIT, ERCIM, Keio, Beihang). W3C liability, trademark and permissive document license rules apply.

Abstract

This module describes the fragmentation model that partitions a flow into pages, columns, or regions. It builds on the Page model module and introduces and defines the fragmentation model. It adds functionality for pagination, breaking variable fragment size and orientation, widows and orphans.

<u>CSS</u> is a language for describing the rendering of structured documents (such as HTML and XML) on screen, on paper, in speech, etc.

Status of this document

https://www.w3.org/TR/css-break-3/#fragmentainer

1/36

8/7/2020

CSS Fragmentation Module Level 3

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at https://www.w3.org/TR/.

This document was produced by the <u>CSS Working Group</u> as a Candidate Recommendation. This document is intended to become a W3C Recommendation. This document will remain a Candidate Recommendation at least until 4 March 2019 in order to ensure the opportunity for wide review.

Multi-columns

•Fragmentainer = column

Fragmentation context =Collection of columns

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper

suscipit lobortis nisl
ut aliquip ex ea
commodo
consequat. Duis
autem vel eum
iriure dolor in
hendrerit in
vulputate velit esse
molestie consequat,
vel illum dolore eu
feugiat nulla
facilisis at vero eros
et accumsan et iusto
odio dignissim qui

blandit praesent
luptatum zzril
delenit augue duis
dolore te feugait
nulla facilisi. Nam
liber tempor cum
soluta nobis
eleifend option
congue nihil
imperdiet doming id
quod mazim
placerat facer
possim assum.

Regions

Fragmentainer = region

Fragmentation context =Collection of regions

rA

I am not a plane Figure, but a Solid. You call me a Circle; but in reality I am not a Circle,

B tan infinite number of Circles, of size varying from a Point to a Circle of thieteen inches in diameter, one placed on the top of the other. When I cut through your plane as I am now doing, I make in your plane a section which you, very rightly, call a Circle. For even a Sphere - which is my

Lorem ipsum dolor areat, consecteir adipiscing elit.
Vivarnas orci rai, feugiat quis feugiat quis feugiat impendier, socierisque et sem. Vivarnas orci di di me rition elementum rutrum sed ac rai. Curabilur rutrum rendem sulfo. di etter rendem sulfo. di etter rendem sulfo. di etter

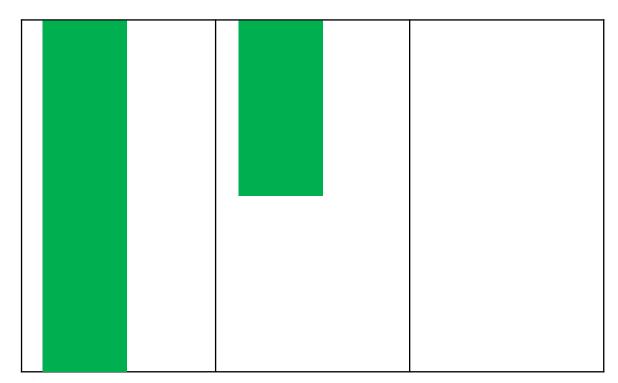
eu consectetur et, pulvinar ut ipsum. Niellam non puras non nibbi allquet tincidunt a nec est. Ut vitae libero eu dolor ullamocopper pharetra. Curabhiar consequat, felis a fringilla varius, lectus turpis auctor enien, id lacinia tellus lacus at sem. Etiam bibendum nibbi vitae purus vestibalam molestie. Phasellas tristique purus eget nisi euismod et fragiat est imperdiet. Phasellus lacreet turpis malesuada nisi sollicitudin scelerisque.

Morbi imperdiet, massa ac molestie lacinia, torior sapien ornare quam, quis lucras nial est et prans. Cras vet odio felis, ut rhoncus urna. Maccenas ut quam eu est vehicula feugiat. Praesent eget leo augue, vet fincidunt mauris. Maccenas accumsan, velit dicum: sagittis imperdiet, elit velit gravida ligula, in pulvinar velit libero ac nibb. Mauris eget rhoncus magna. Donce ultricies massa untor, nec cuismod nibb. Sed a nibb mauris, venenatis gravida nial. Cras neque metus, vulputate et cursus non, ultricies sit arnet metus. Praesent vestibulum lacinia gravida. Nunc eget mulla quis sellos consectetar aliquam in quis praus. Sed nec turpis vel leo vehicula vulputate ac at diam. Acessa at turpis non enim viverra faucibus a in massa. Donce et dui a leo scelerisque luctus. Cras in justo a mauris bendrerit

proper name in my own country - if he manifest himself at all to an inhabitant of Flatland - must needs manifest himself as a Circle. Square and triangular houses are not allowed, and for this reason. The angles of a Square (and still more those of an equilateral Triangle.) being much more pointed than those of a Pentagon, and the lines of inanimate objects (such as houses) being dimmer than the lines of Men and Women. it follows that there is no little danger lest the points of a square or triangular house residence might do serious injury to an inconsiderate or perhaps absent-minded traveller suddenly therefore, running against them; and as early as the eleventh century of our era, triangular houses were universally forbidden by Law, the only exceptions being fortifications, powdermagazines, barracks, and other state buildings, which it is not desirable that the general public should approach without circumspection.

- Fragment
 - "The portion of a box that belongs to exactly one <u>fragmentainer</u>."

- Fragment
 - "The portion of a box that belongs to exactly one <u>fragmentainer</u>."



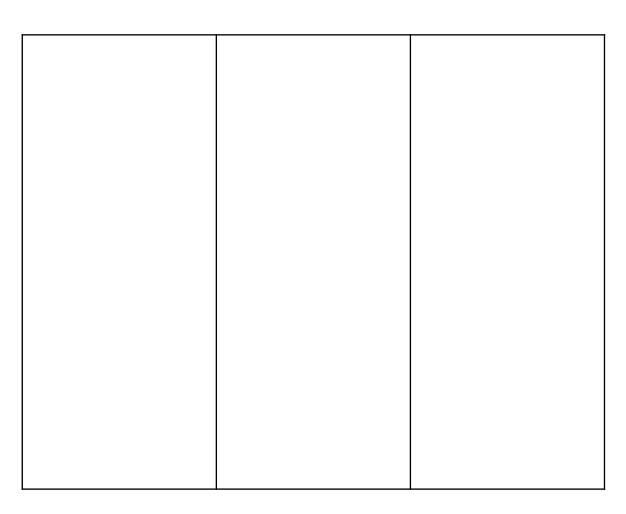
- Fragment
 - "The portion of a box that belongs to exactly one <u>fragmentainer</u>."
- Monolithic content
 - Content that isn't allowed to fragment
 - •Ex: images and videos

CSS Specifications

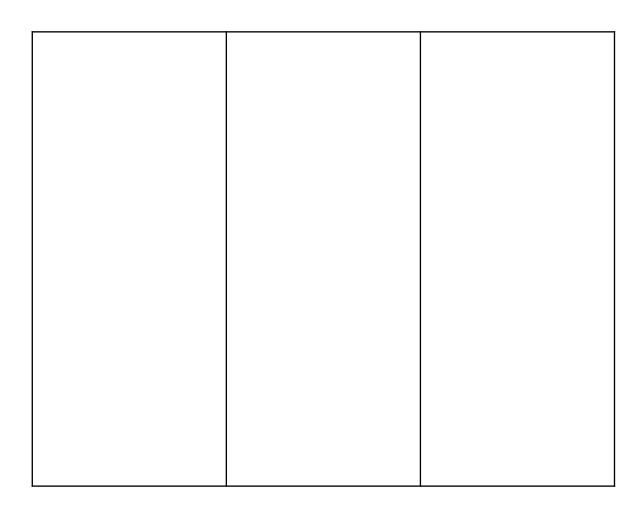
- CSS Fragmentation Module Level 3
 (https://drafts.csswg.org/css-break/)
- CSS Multi-column Layout Module Level 1 (https://drafts.csswg.org/css-multicol/)
- CSS Paged Media Module Level 3
 (https://drafts.csswg.org/css-page-3/)
- CSS Regions Module Level 1
 (https://drafts.csswg.org/css-regions/)

Fragmentation in LayoutNG

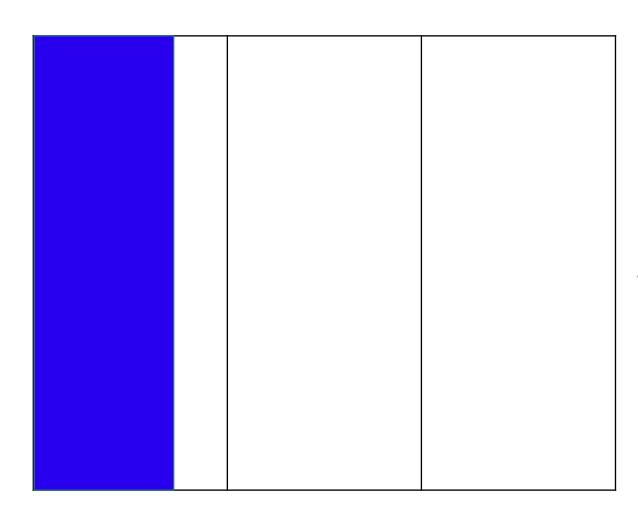
```
.multicol {
  column-count: 3;
  height: 100px;
}
```



```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



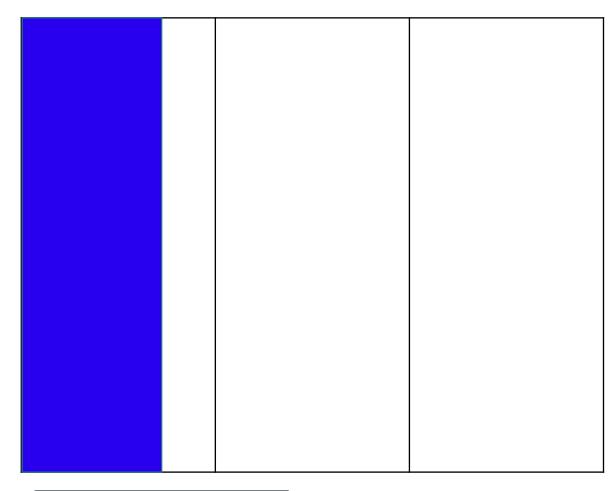
```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



BreakToken

```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```

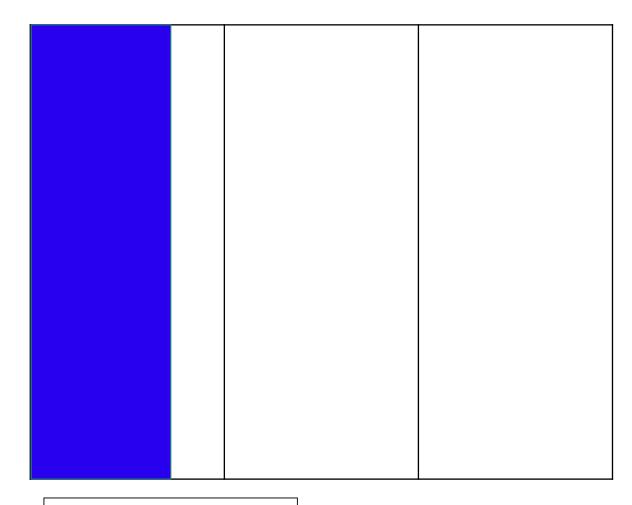
```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



consumed_block_size

BreakToken

```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```

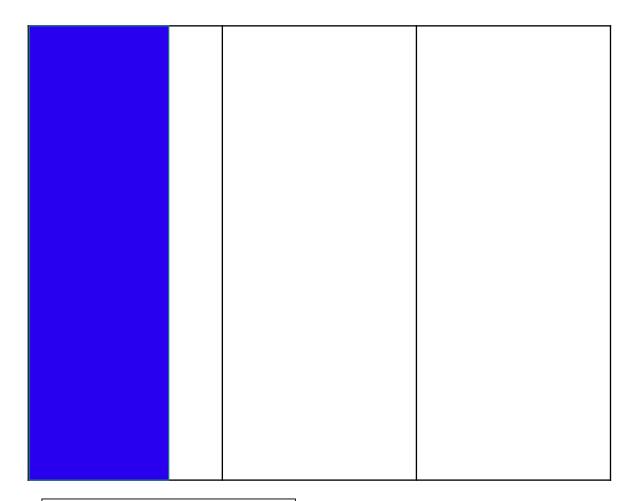


100px

consumed_block_size=100

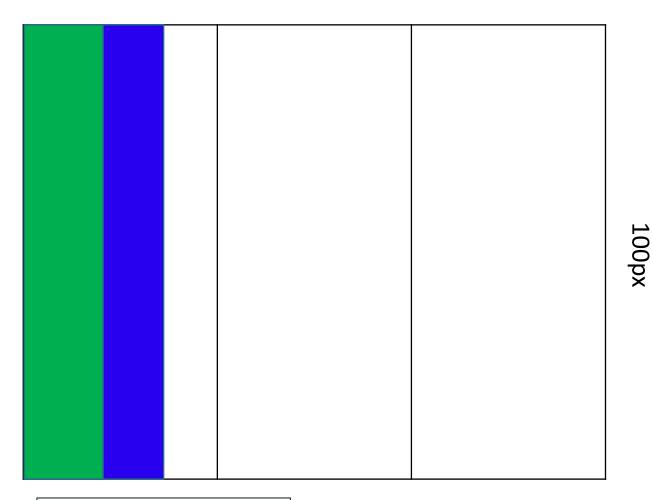
BreakToken

```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```

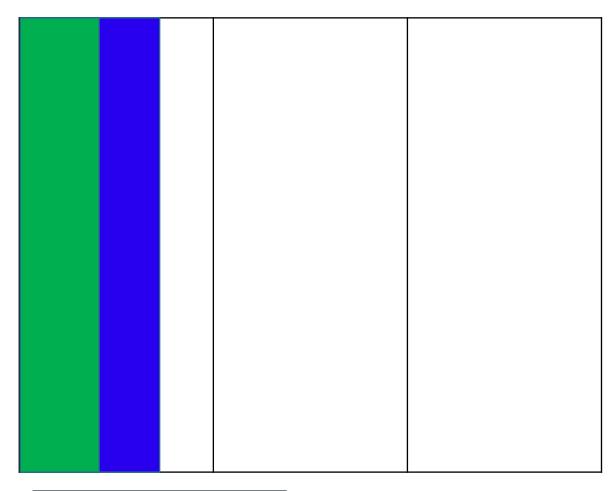


100px

```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```

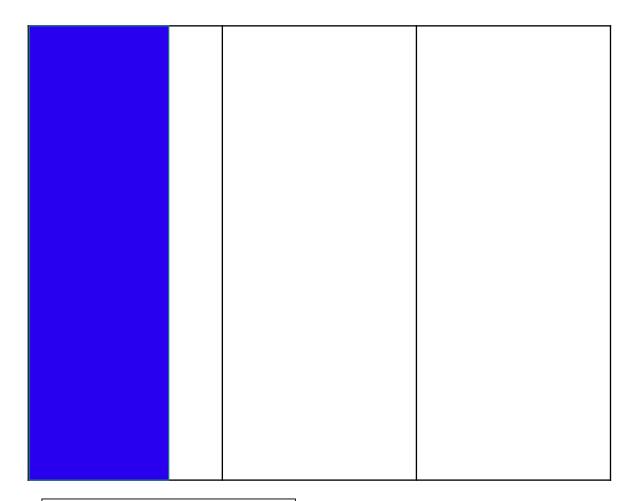


```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



100px

```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```

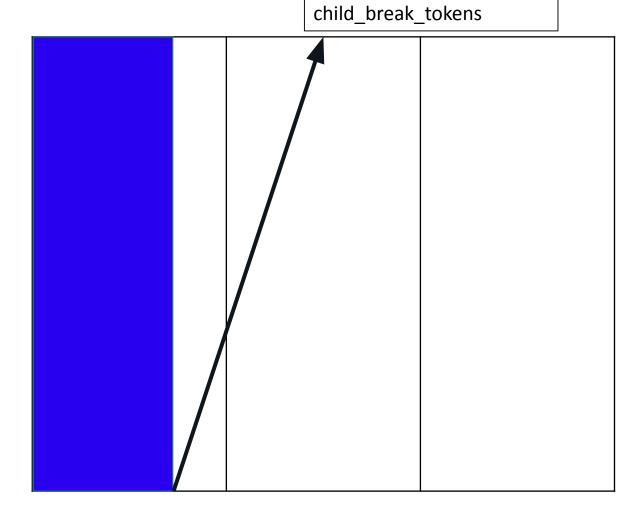


100px

100px

Multi-column Example

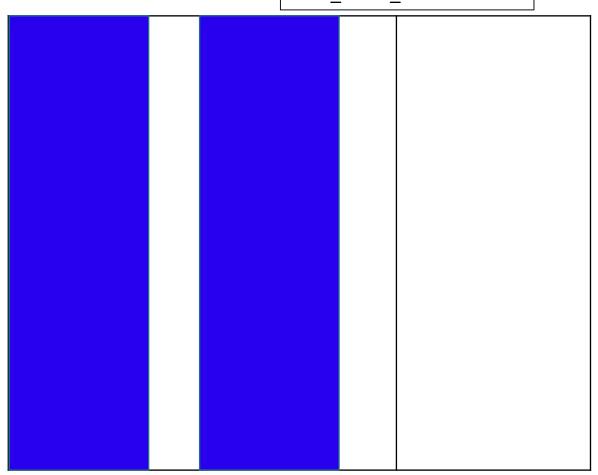
```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



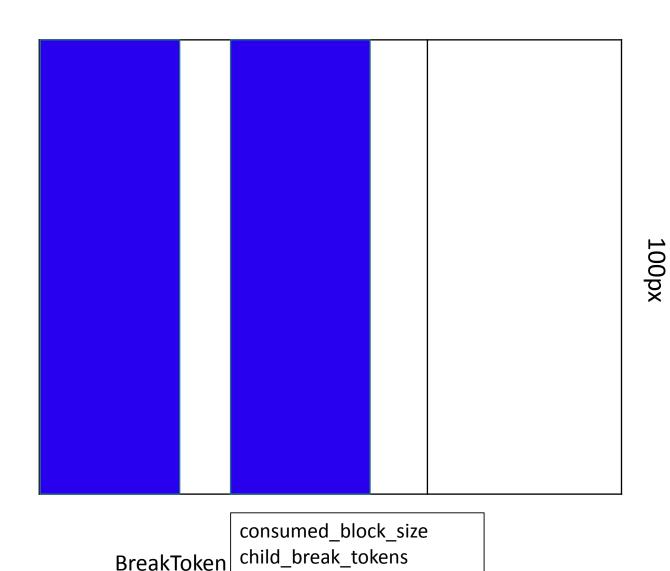
consumed_block_size=100

```
consumed_block_size=100 child_break_tokens
```

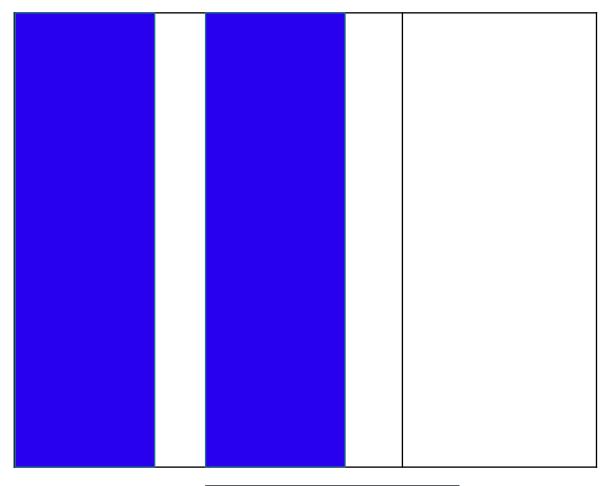
```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



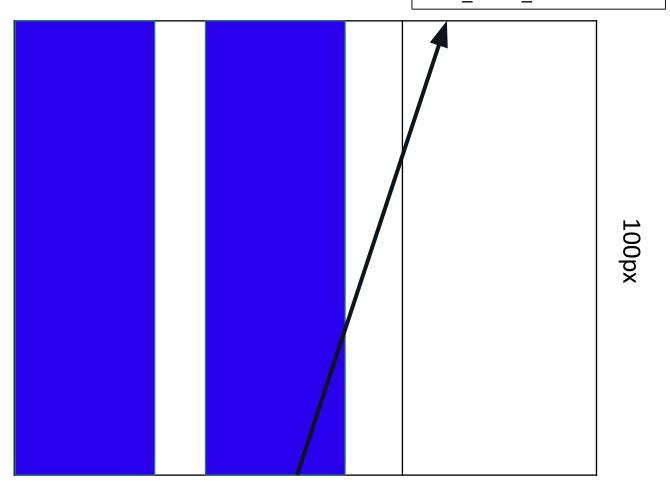
```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



100px

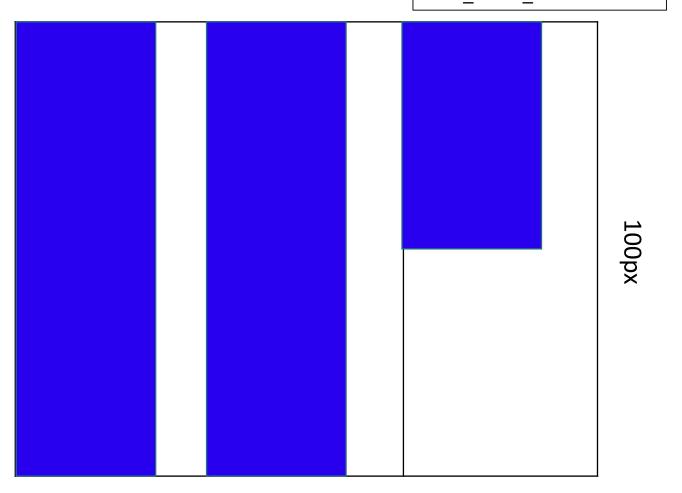
consumed_block_size=200
child_break_tokens

```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```

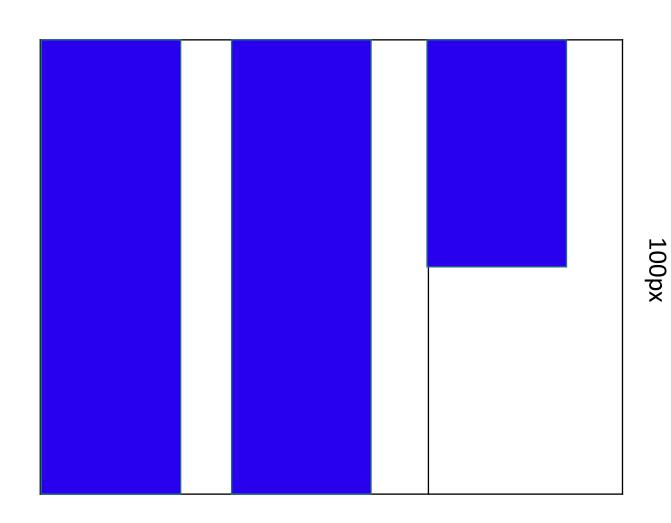


```
consumed_block_size=200
child_break_tokens
```

```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



```
.multicol {
 column-count: 3;
 height: 100px;
#child {
 height: 250px;
 background: blue;
```



Out-of-flow Positioned Overview

CSS position property

- static
- relative
- fixed
- absolute
- sticky

CSS position property

- static
- relative
- fixed out-of-flow out-of-flow
- sticky

CSS position property

- static
- relative
- fixed out-of-flow out-of-flow
- sticky

'position: absolute'

- Taken out of flow
- Positioned relative its containing block
 - i.e. Relative to its nearest positioned ancestor

'position: absolute'

- Taken out of flow
- Positioned relative its containing block
 - i.e. Relative to its nearest positioned ancestor

'position: absolute'

- Taken out of flow
- Positioned relative its containing block
 - i.e. Relative to its nearest positioned ancestor

Out-of-flow Positioned in LayoutNG

- Block node
 - The out-of-flow node that we need to perform layout on
- Static position
 - The offset the out-of-flow node had it been in-flow

Out-of-flow Positioned Fragmentation

```
<div style="columns: 3;">
                                        <u>OutOfFlowPositionedNode</u>
  <div style="position: relative;">
     <div>
       <div style="position: absolute; bottom:0;"></div>
     </div>
  </div>
</div>
```

```
<div style="columns: 3;">
                                        <u>OutOfFlowPositionedNode</u>
  <div style="position: relative;">
     <div>
       <div style="position: absolute; bottom:0;"></div>
     </div>
  </div>
</div>
```

```
<div style="columns: 3;">
                              <u>OutOfFlowPositionedNode</u>
  <div style="position: relative;">
     <div>
       <div style="position: absolute; bottom:0;"></div>
     </div>
  </div>
</div>
```

```
<u>OutOfFlowPositionedNode</u>
<div style="columns: 3;">
  <div style="position: relative;">
     <div>
       <div style="position: absolute; bottom:0;"></div>
     </div>
  </div>
</div>
```

Step 1: Bubble nodes up to fragmentation context root

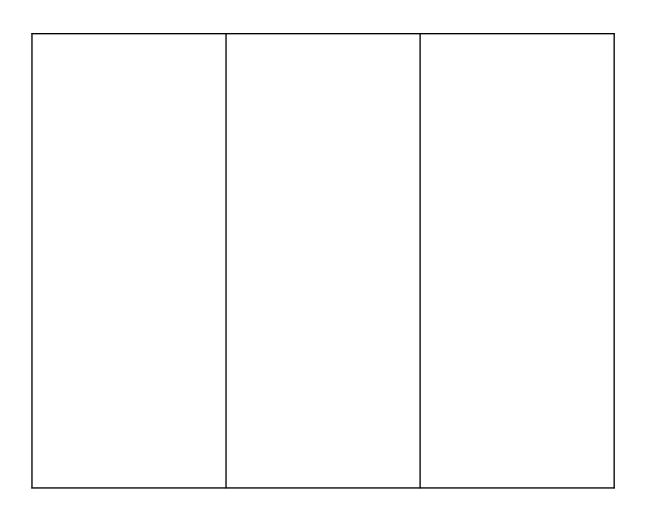
- We are performing layout at a later point in time
- We need more information to position correctly
 - What is the containing block?
 - Where is the containing block located?

Step 1: Bubble nodes up to fragmentation context root

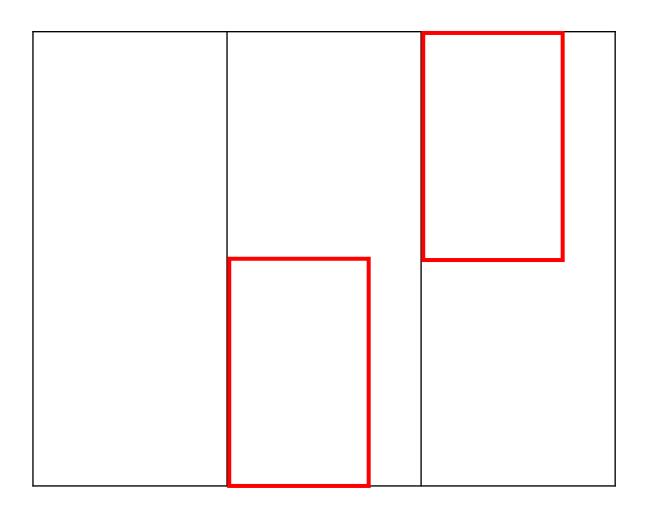
Collecting extra info in OutOfFlowPositionedNode

- Block node
 - The out-of-flow node that we need to perform layout on
- Static position
 - The offset the out-of-flow node had it been in-flow

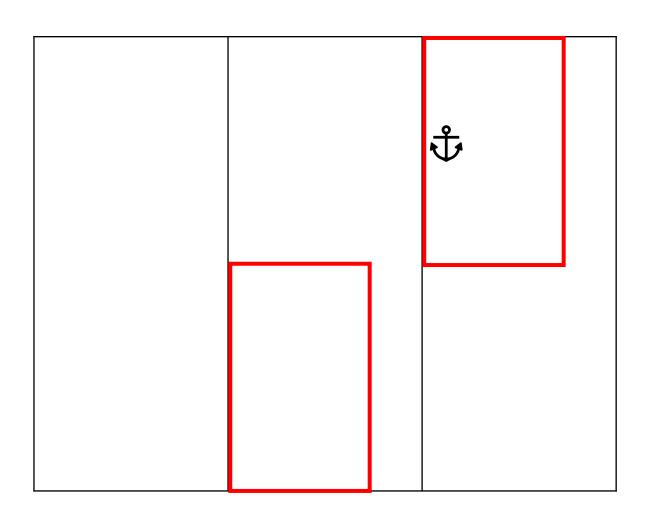
- Block node
 - The out-of-flow node that we need to perform layout on
- Static position
 - The offset the out-of-flow node had it been in-flow
- Containing block physical fragment
 - Reference to the containing block we are positioned relative to
- Containing block offset
 - The offset from the first containing block fragment to the fragmentation context root (as if all fragmentainers were lined up in the block direction)



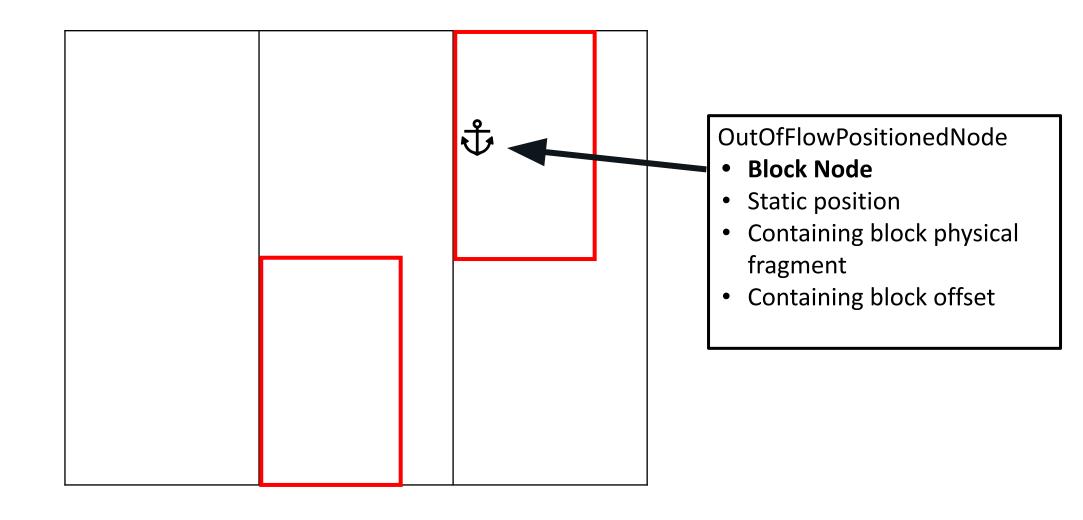
- Block Node
- Static position
- Containing block physical fragment
- Containing block offset

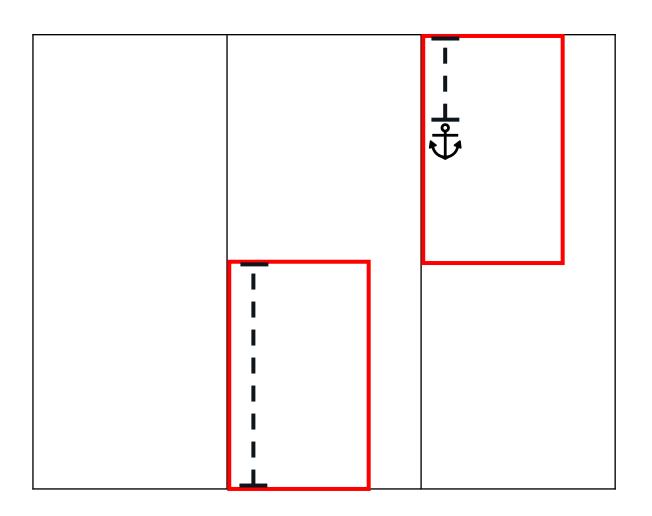


- Block Node
- Static position
- Containing block physical fragment
- Containing block offset

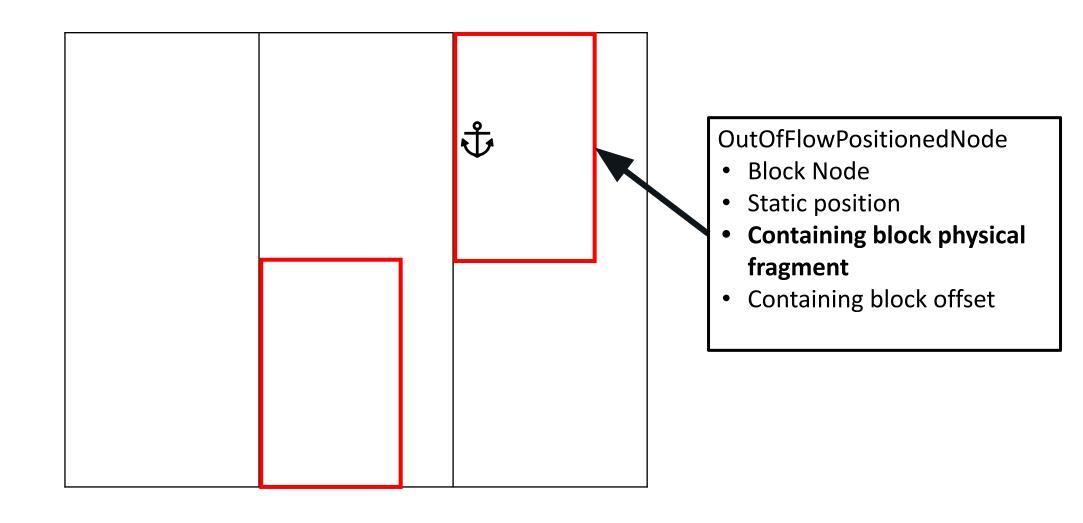


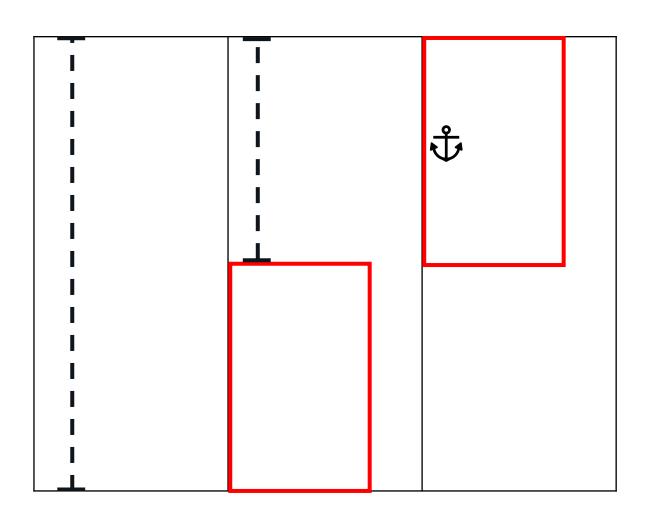
- Block Node
- Static position
- Containing block physical fragment
- Containing block offset





- Block Node
- Static position
- Containing block physical fragment
- Containing block offset





- Block Node
- Static position
- Containing block physical fragment
- Containing block offset

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

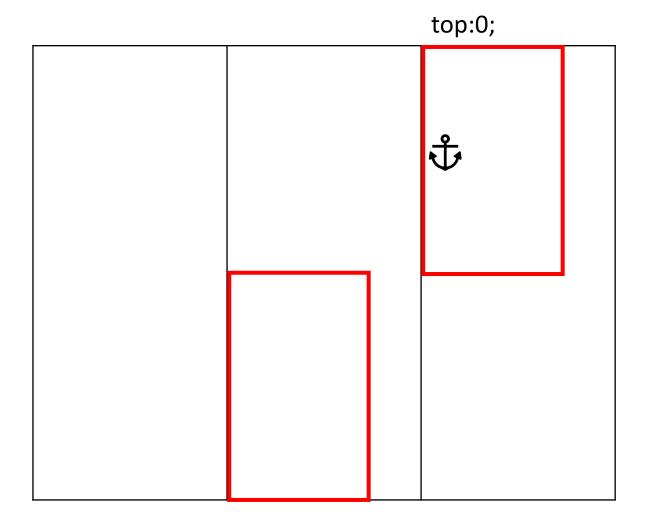
Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

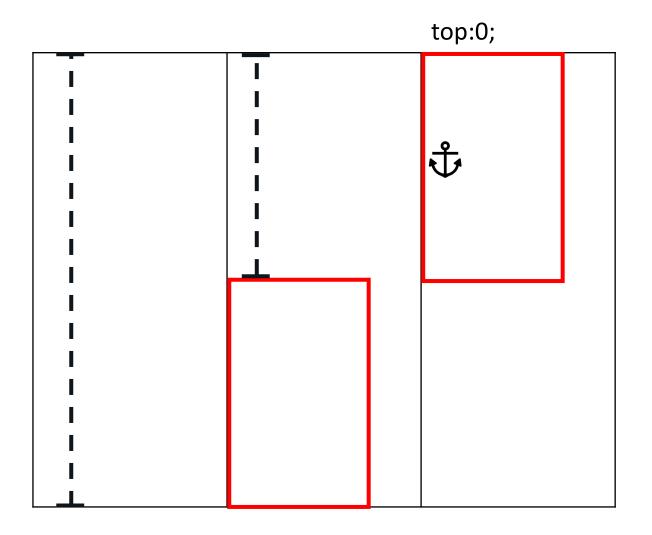
Start offset

Start Offset



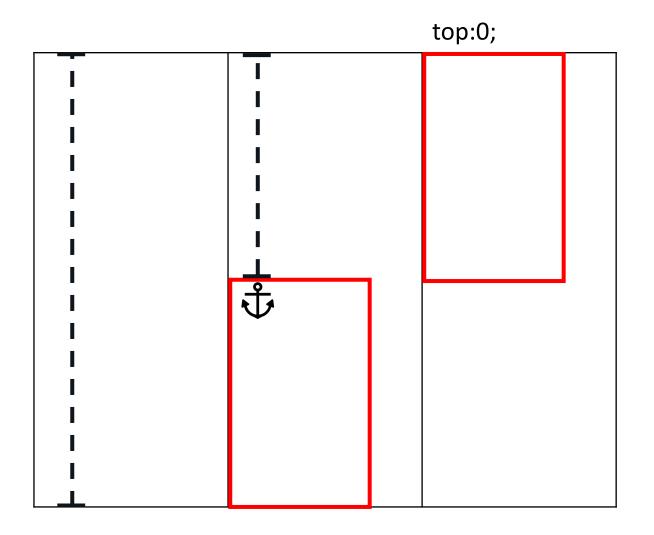
- Block Node
- Static position
- Containing block physical fragment
- Containing block offset

Start Offset



- Block Node
- Static position
- Containing block physical fragment
- Containing block offset

Start Offset



- Block Node
- Static position
- Containing block physical fragment
- Containing block offset

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

Start offset

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

- Start offset
- Fragmentainer to start layout

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

- Start offset
- Fragmentainer to start layout
- Constraint space

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

- Start offset
- Fragmentainer to start layout
- Constraint space
- Layout() -> LayoutResult

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

- Start offset
- Fragmentainer to start layout
- Constraint space
- Layout() -> LayoutResult
- Handle break tokens

Handle Break Tokens

- If the LayoutResult produced BreakToken
 - Determine the next fragmentainer
 - Create a new ConstraintSpace
 - Perform Layout
 - Repeat until no BreakToken

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

- Start offset
- Fragmentainer to start layout
- Constraint space
- Layout() -> LayoutResult
- Handle break tokens

Step 3: Transplant the OOF fragments into fragmentainers

Issue #3

- We want to add the OOF fragments to fragmentainers
- Fragmentainers have already finished layout
- PhysicalFragments are immutable

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

- Start offset
- Fragmentainer to start layout
- Constraint space
- Layout() -> LayoutResult
- Handle break tokens

Step 3: Transplant the OOF fragments into fragmentainers

Clone fragmentainer

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

- Start offset
- Fragmentainer to start layout
- Constraint space
- Layout() -> LayoutResult
- Handle break tokens

Step 3: Transplant the OOF fragments into fragmentainers

- Clone fragmentainer
- Add OOF children

Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

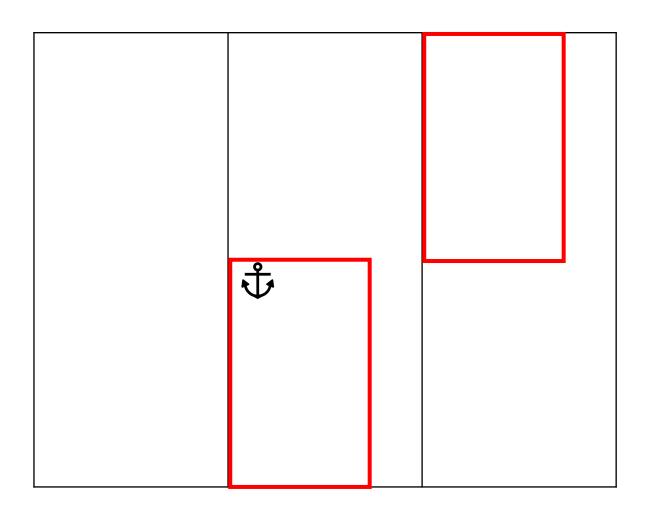
- Start offset
- Fragmentainer to start layout
- Constraint space
- Layout() -> LayoutResult
- Handle break tokens

Step 3: Transplant the OOF fragments into fragmentainers

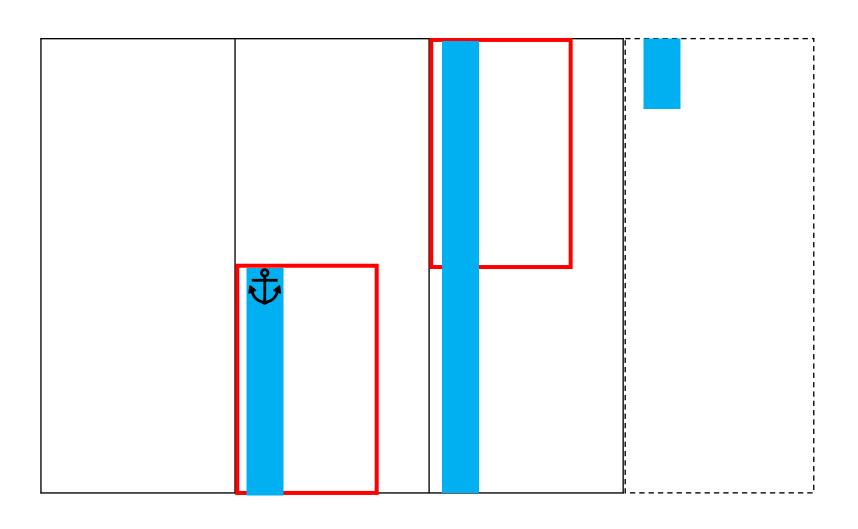
- Clone fragmentainer
- Add OOF children

Step 4: Add new fragmentainers

Add New Fragmentainers



Add New Fragmentainers



Step 1: Bubble nodes up to fragmentation context root

Collecting extra info in OutOfFlowPositionedNode

Step 2: Layout the OOF positioned nodes

- Start offset
- Fragmentainer to start layout
- Constraint space
- Layout() -> LayoutResult
- Handle break tokens

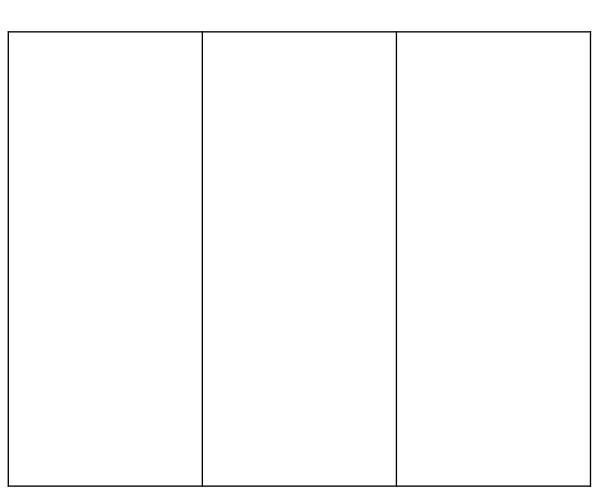
Step 3: Transplant the OOF fragments into fragmentainers

- Clone fragmentainer
- Add OOF children

Step 4: Add new fragmentainers

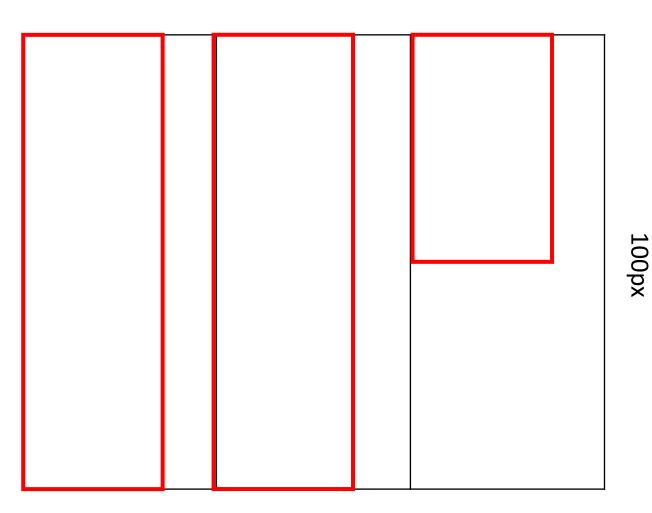


```
.multicol {
  column-count: 3;
  height: 100px;
}
```

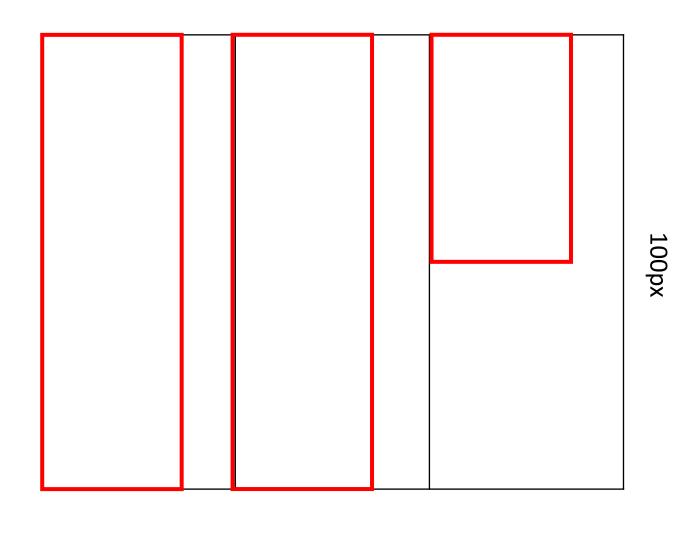


100px

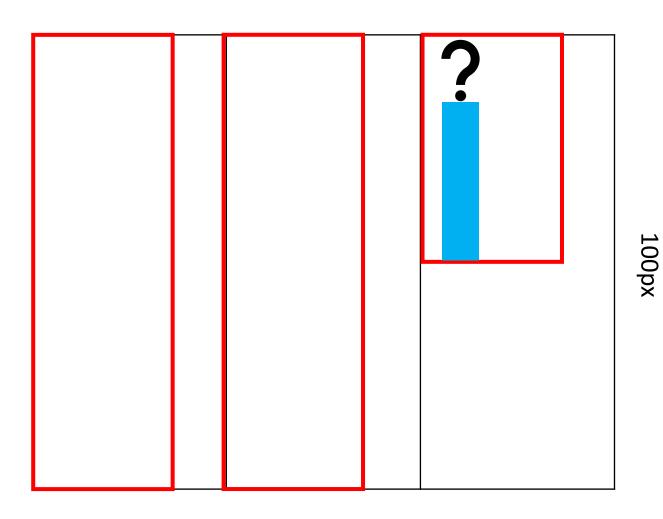
```
.multicol {
 column-count: 3;
 height: 100px;
#rel {
 position: relative;
 height: 250px;
 outline: solid red;
```



```
.multicol {
 column-count: 3;
 height: 100px;
#rel {
 position: relative;
 height: 250px;
 outline: solid red;
#abs {
 position: absolute;
 height: auto;
 bottom: 0;
```



```
.multicol {
 column-count: 3;
 height: 100px;
#rel {
 position: relative;
 height: 250px;
 outline: solid red;
#abs {
 position: absolute;
 height: auto;
 bottom: 0;
```



- Run an additional non-fragmenting layout pass
- This will give us the block size -> start offset
- Then we can perform layout

```
<u>OutOfFlowPositionedNode</u>
<div style="columns: 3;">
  <div style="position: relative;">
     <div>
       <div style="position: absolute; bottom:0;"></div>
     </div>
  </div>
</div>
```

```
<div style="columns: 2;">
                                <u>OutOfFlowPositionedNode</u>
  <div style="columns: 3;">
    <div style="position: relative;">
       <div>
         <div style="position: absolute; bottom:0;"></div>
       </div>
    </div>
  </div>
</div>
```

```
<div style="columns: 2;">
                                <u>OutOfFlowPositionedNode</u>
  <div style="columns: 3;">
    <div style="position: relative;">
       <div>
         <div style="position: absolute; bottom:0;"></div>
       </div>
    </div>
  </div>
</div>
```

```
<u>OutOfFlowPositionedNode</u>
<div style="columns: 2;">
  <div style="columns: 3;">
    <div style="position: relative;">
       <div>
         <div style="position: absolute; bottom:0;"></div>
       </div>
    </div>
  </div>
</div>
```

```
<u>OutOfFlowPositionedNode</u>
<div style="columns: 2;">
  <div style="columns: 3;">
    <div style="position: relative;">
       <div>
         <div style="position: absolute; bottom:0;"></div>
       </div>
    </div>
  </div>
</div>
```

Design TBD

Contributing Team

Rossen Atanassov Benjamin Beaudry Morten Stenshorne Ian Kilpatrick Alison Maher Daniel Libby

