pxCode 開發經驗與技術分享

Modern Web 2021

Chris Chang

Chris Chang

About Me

- Ulead / Corel / HTC / Cheetah Mobile
- MiracleMobile / MonChats Founder
- pxCode Founder
- 熱愛寫程式、軟體、看書、音樂

- Facebook 歡迎加好友
  - https://www.facebook.com/reality.chang

大綱

- 緣起 - pxCode
- 切版的思考模型
- 更有彈性的適配定位 - **Posize**
- 不跑版的 Bootstrap - **Fusion Flex**
- 總結

緣起

- 我是一家軟體服務公司的創辦人
  - 案子好多，時程好趕，人力不足
  - 切版好辛苦好慢、需要耐心、溝通、適配、瀏覽器與裝置測試
  - 來來回回溝通、討論
  - 設計師常覺得工程師，切版都會跑版
  - 工程師常覺得設計師，設計都沒先想清楚響應式行為

★ 有沒有一個好的方案，可以解決切版這個問題？

萬里長征 的 開始

- 解決方案的目標
  - 直接匯入設計稿 (Sketch / Figma)
  - 切版要夠快，至少十倍速 → 10x Faster
  - 不能跑版 → Pixel Perfect
  - 程式碼易讀、好整合功能 → Clean Code

- 天真的我
  - 跟團隊說，我們三個月就出一版吧！
  - 每次的迭代，永遠自我懷疑還有多少不懂的 HTML 與 CSS

行百里路 的 現況

- 24 個月 的 迭代與優化
  - 「切版命題」超級困難，是超大坑裡，又有數不盡的小坑
  - 歷經超過 50 個版本的迭代
  - 30 場以上線下用戶的使用者測試與訪談
  - 全球目前近萬線上用戶的使用與回饋
  - → 正在一點一滴地修正與優化
- 目前成果
  - 不跑版 Pixel Perfect
  - 速率從 5x → 10x → 20x
  - 程式碼也愈來愈簡潔乾淨，而且可以直切響應式

pxCode

歡迎試用

- pxCode YouTube
- pxCode Medium
- https://pxcode.io

分享 累積的技術經驗

- 切版思考模型
  - 簡單的幾個技巧，可以讓不會切版的人，也能在幾天 內上手開始切版
- Posize
  - 更有彈性的適配定位
- Fusion Flex
  - 魚與熊掌兼得，不會跑版的 Bootstrap

切版思考模型

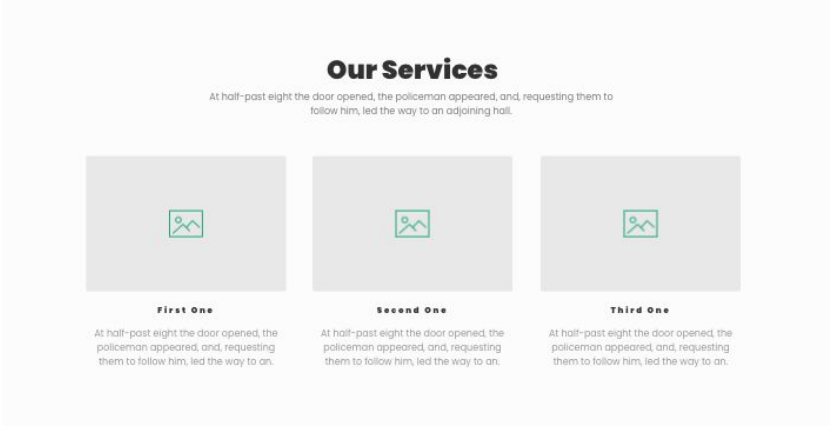
目標

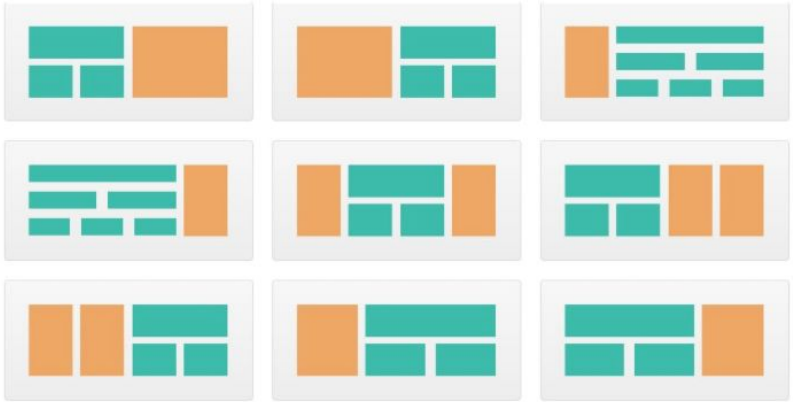
- 嘗試找出一個簡易的切版思考解題模型
  - 讓切版使用者更輕鬆
  - 讓不會切版的人，也能快速學習上手
  - 盡可能讓編輯變的更簡單，效率變的更好

- 可以用這個簡單的模型，解決 90% 以上的切版問題
  - 尋找 500+ Sketch / Figma 的視覺稿作驗証

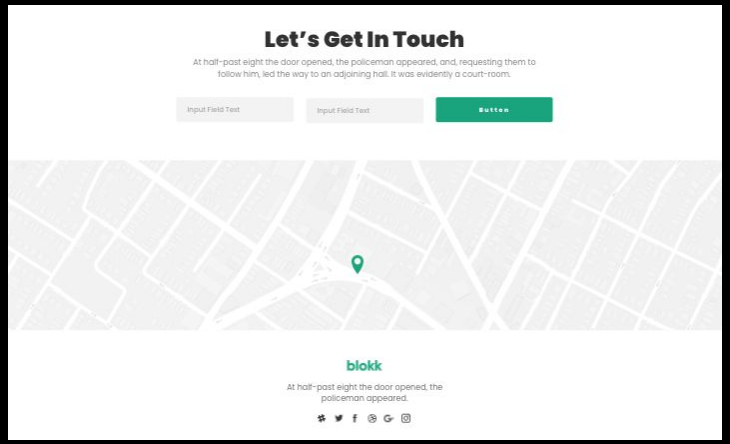
網頁中的 主要內容

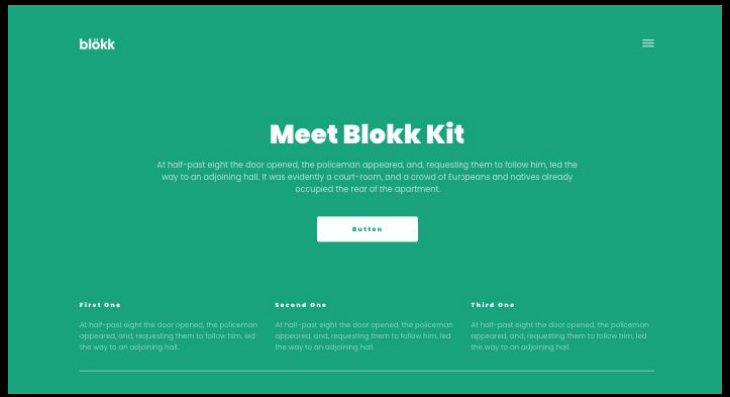
- 網頁內容，先區分為「主要內容」與「附屬內容」
  - 通常**主要內容**是「直排」與「橫排」的排列組合

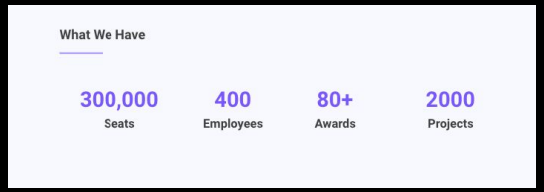


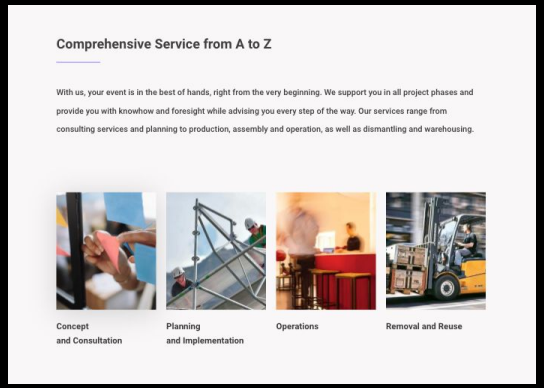


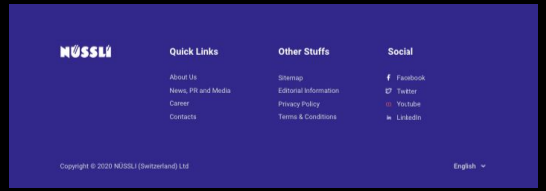
More Examples - 網頁就是「直直橫橫」的「排列組合」

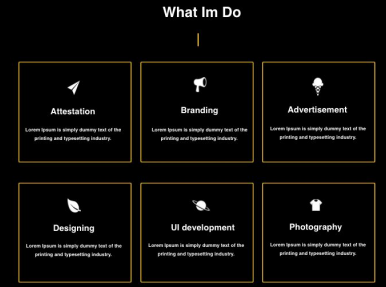


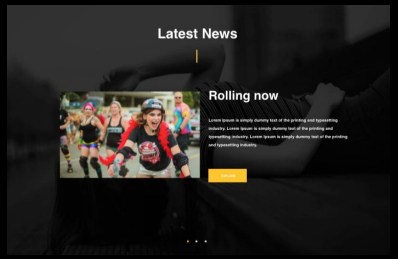


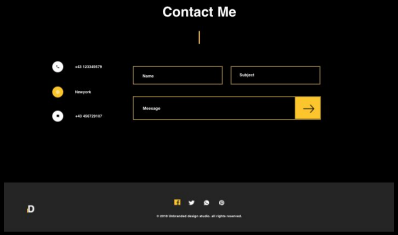




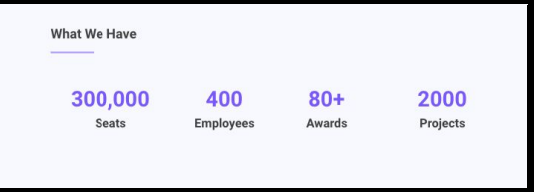


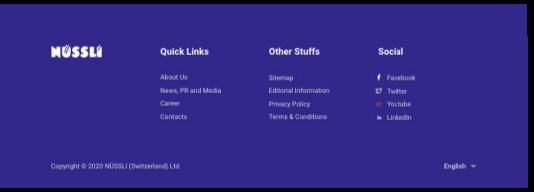


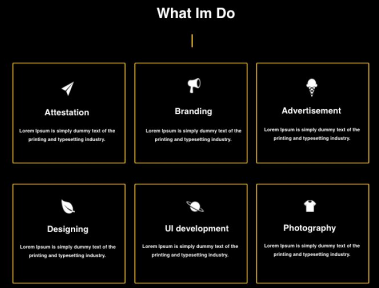


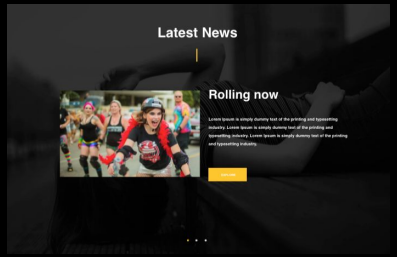


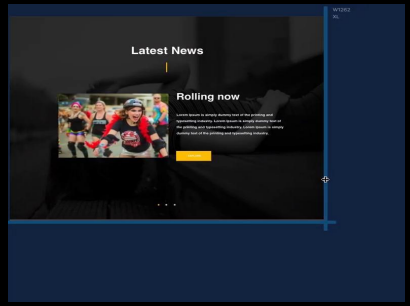
More Examples - 網頁就是直直橫橫的流動內容

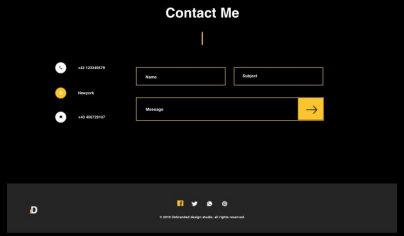






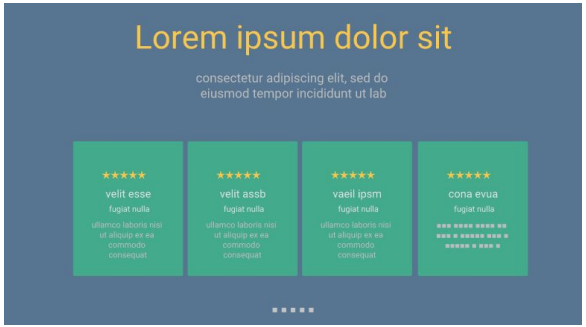


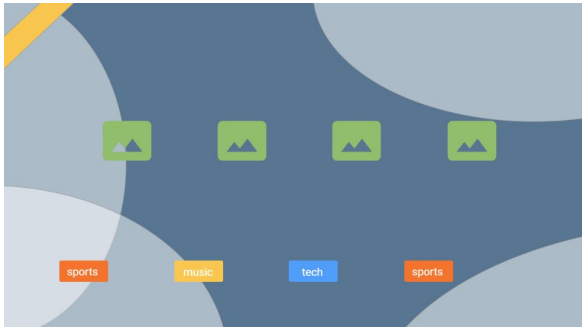




網頁的 附屬內容

- 附屬內容，選擇依附在某一個主要內容上





3 Basic **HTML Group** in pxCode

FlexGroup Column

```
<div class='flex column'>
    <div class='item1'></div>
    <div class='item2'></div>
    <div class='item3'></div>
</div>
```

FlexGroup Row

```
<div class='flex row'>
    <div class='item1'></div>
    <div class='item2'></div>
    <div class='item3'></div>
</div>
```

Normal Group

```
<div class='relative'>
    <div class='absolute fg-item'></div>
    <div class='main_child'></div>
    <div class='absolute bg-item'></div>
</div>
```

3 Basic **HTML Group** in pxCode

FlexGroup Column

```
<div class='flex column'>
    <div class='item1'></div>
    <div class='item2'></div>
    <div class='item3'></div>
</div>
```

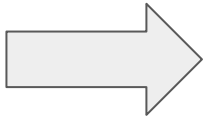
FlexGroup Row

```
<div class='flex row'>
    <div class='item1'></div>
    <div class='item2'></div>
    <div class='item3'></div>
</div>
```

Normal Group

```
<div class='relative'>
    <div class='absolute fg-item'></div>
    <div class='main_child'></div>
    <div class='absolute bg-item'></div>
</div>
```

可排列組合出不同的設計



3 Basic **HTML Group** in pxCode

FlexGroup Column

```
<div class='flex column'>
    <div class='item1'></div>
    <div class='item2'></div>
    <div class='item3'></div>
</div>
```

FlexGroup Row

```
<div class='flex row'>
    <div class='item1'></div>
    <div class='item2'></div>
    <div class='item3'></div>
</div>
```

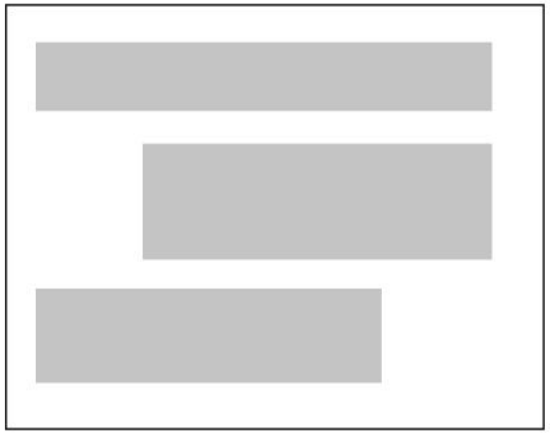
Normal Group

```
<div class='relative'>
    <div class='absolute fg-item'></div>
    <div class='main_child'></div>
    <div class='absolute bg-item'></div>
</div>
```

可排列組合出不同的設計

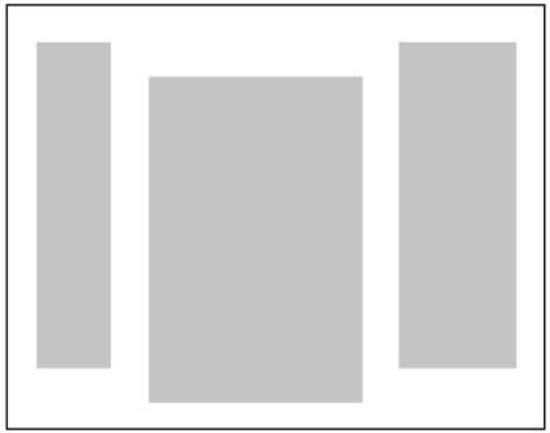
將設計稿 編輯成 結構化的 HTML

pxCode 模型總結



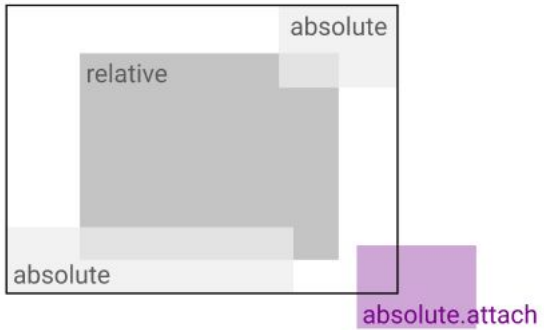
Flex Column 直排

- div.flex_column
  - div
  - div



Flex Row 橫排

- div.flex_row
  - div
  - div



Group 群組

- div.group
  - div.relative 主要子內容
  - div.absolute 子內容
  - div.absolute.attach 附屬子內容

思考切版的 SOP

- 先找出**主要內容**, 忽略附屬內容
- 想辦法排直的 或是 排橫的
- 排列組合成最終的結果
- 回頭再一個個附屬內容, 貼回適合的主要內容
- 加入響應式行為

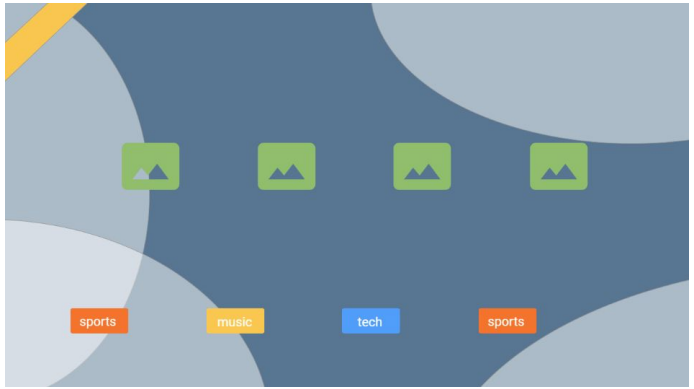
讓我們練習用嘴巴切版

Lorem ipsum dolor sit
consectetur adipiscing elit, sed
eiusmod tempor incididunt ut lab
velit esse
fugiat nulla
ullamco laboris nisi ut aliquip ex ea commodo consequat
velit assb
fugiat nulla
ullamco laboris nisi ut aliquip ex ea commodo consequat
vaeil ipsm
fugiat nulla
ullamco laboris nisi ut aliquip ex ea commodo consequat
cona evua
fugiat nulla
sports
music
tech
sports

先找出主要內容，隱藏附屬內容

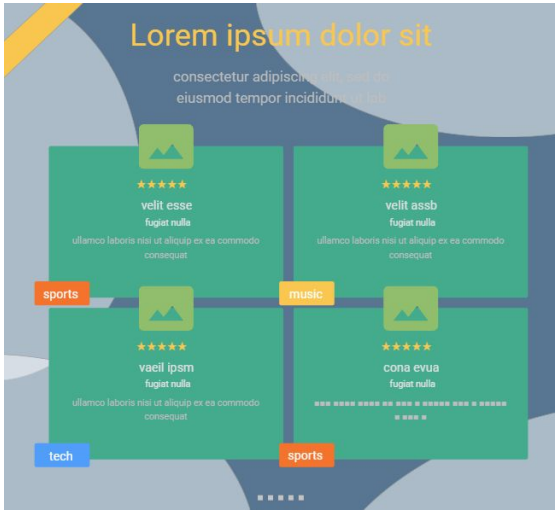
結構化主要內容 - 直排與橫排的排列組合

附屬內容 各自選擇依附的 主要內容



加入響應式行為

Lorem ipsum dolor sit
consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut lab
velit esse
fugiat nulla
ullamco laboris nisi ut aliquip ex ea commodo consequat
velit assb
fugiat nulla
ullamco laboris nisi ut aliquip ex ea commodo consequat
vaeil ipsm
fugiat nulla
ullamco laboris nisi ut aliquip ex ea commodo consequat
cona evua
fugiat nulla
sports
music
tech
sports

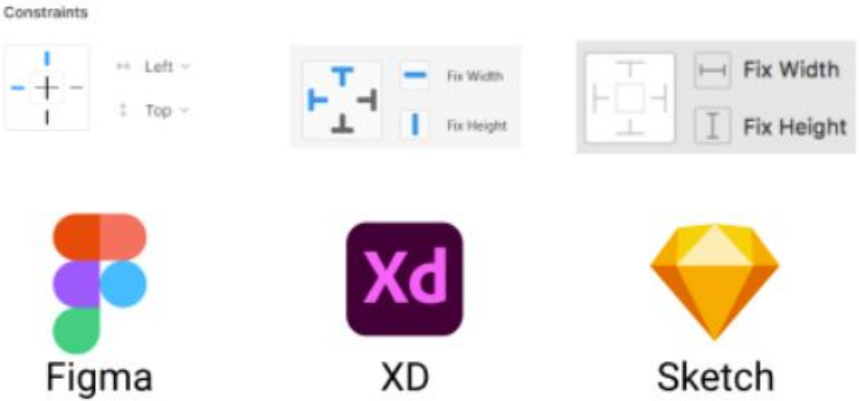

Lorem ipsum dolor sit
velit esse
fugiat nulla
velit assb
fugiat nulla
vaeil ipsm
fugiat nulla
cona evua
fugiat nulla
sports
music
tech
sports

Lorem ipsum dolor sit
velit esse
fugiat nulla
velit assb
fugiat nulla
vaeil ipsm
fugiat nulla
cona evua
fugiat nulla
sports
music
tech
sports

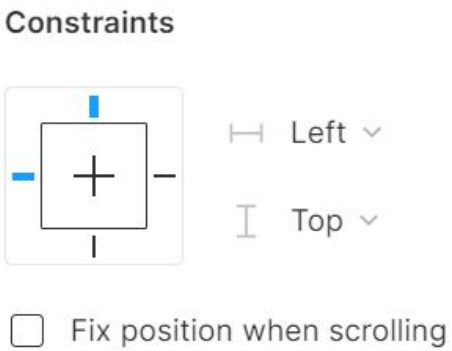
Posize

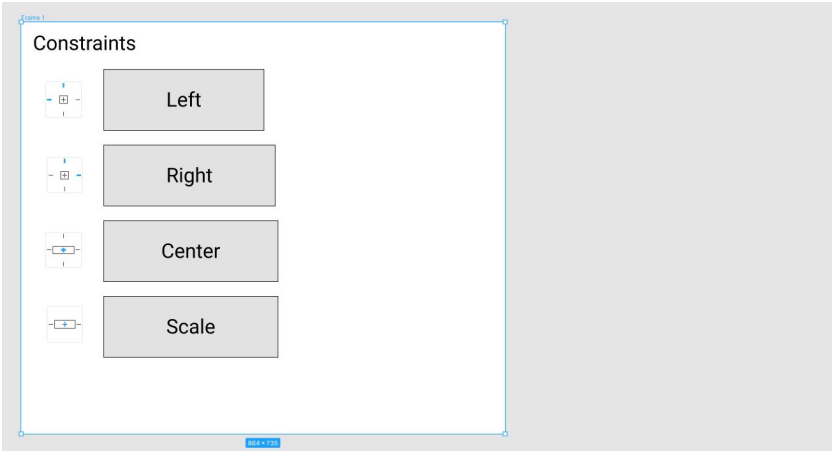
Posize 是從那來的？

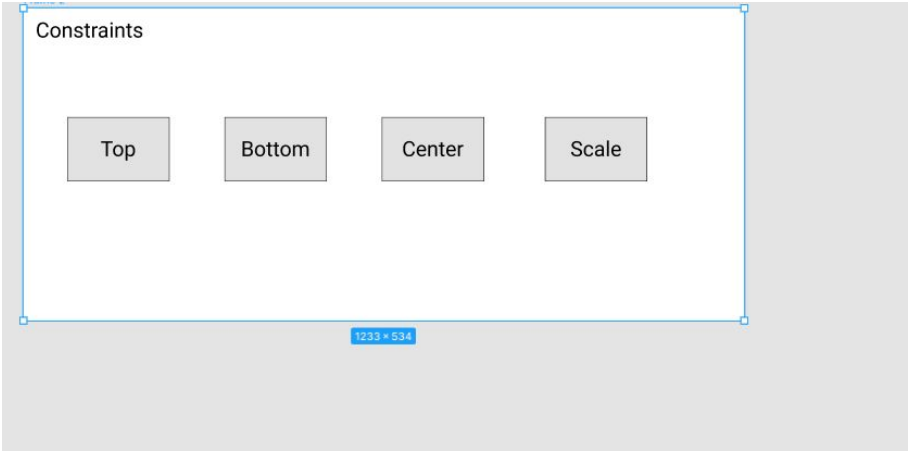
- 設計 **響應式** 網頁
  - 所有設計軟體，皆有 Resizing Contraints 的功能
  - 讓設計師能容易設計出不同解析度的版面設計



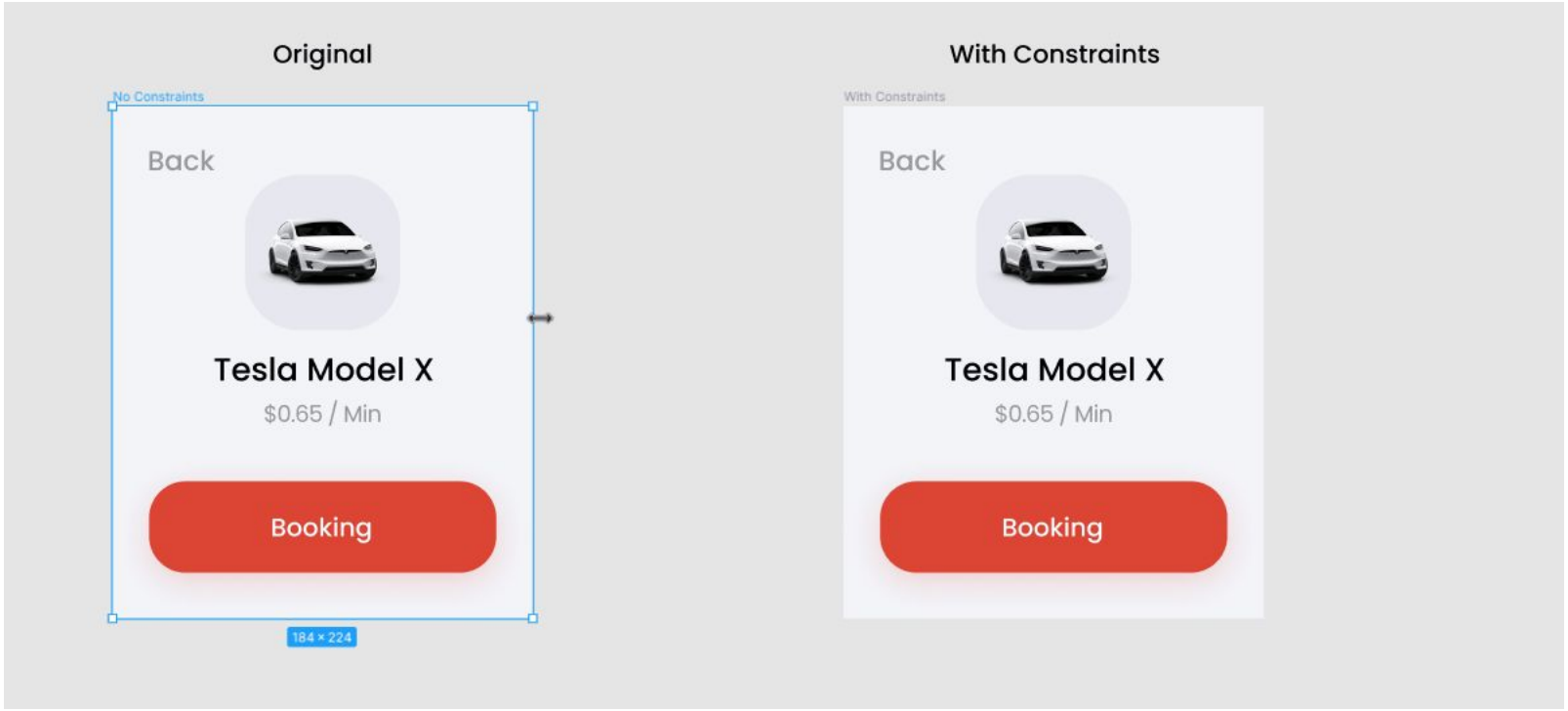
Constraints 演示 in Figma





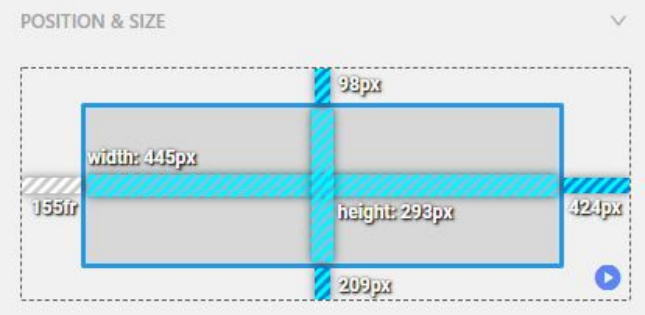


Constraints 演示 in Figma

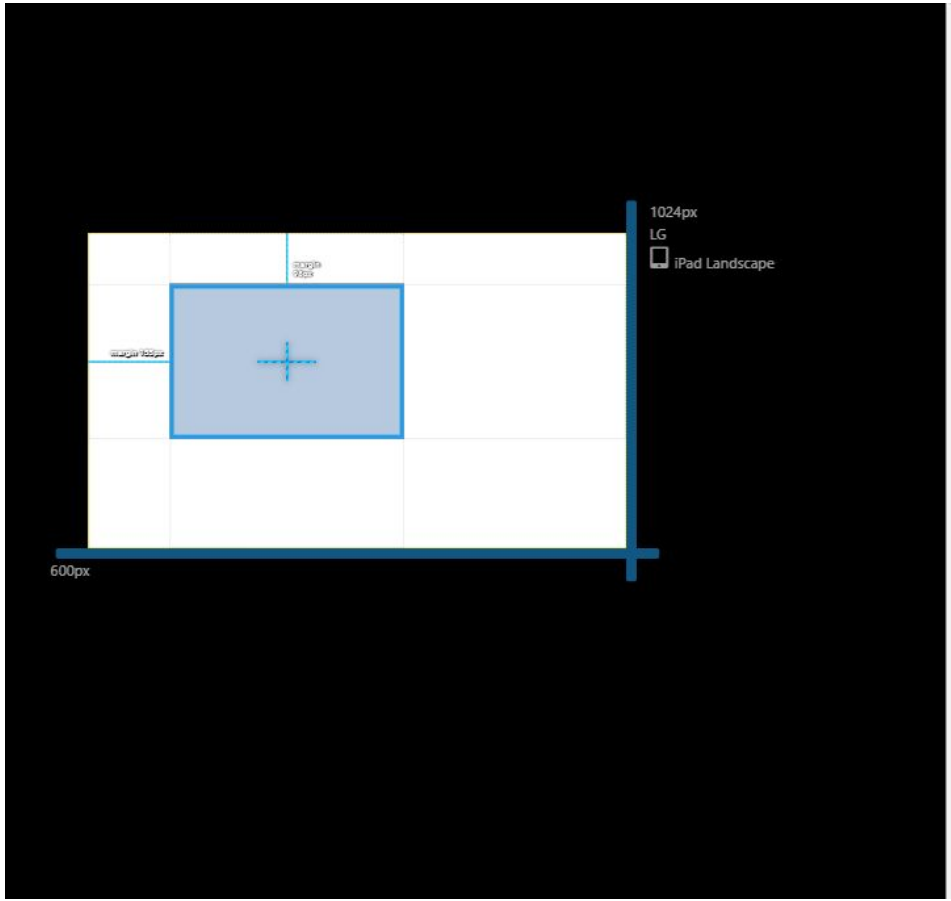


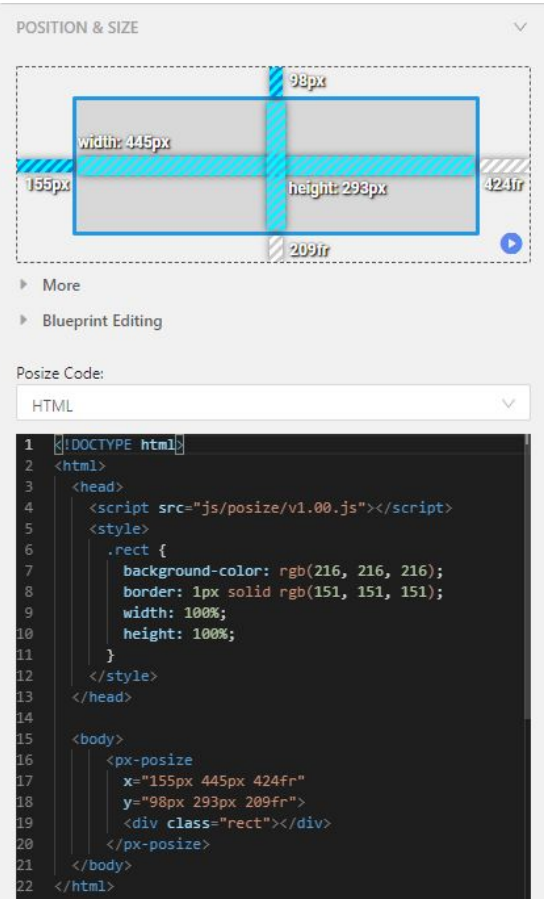
Posize

- **Posize**
  - 實作了 **Resizing Constraints**
  - 命名來自 Position + Size
  - 雙十字的控制項
  - 外十字 Position - 定義**定位適配**行為
  - 內十字 Size (Width + Height) - 定義**大小適配**行為
- 開源 Posize 實作
  - https://github.com/px2code/posize HTML 版本
  - https://github.com/px2code/react-posize React 版本



Posize Playground 演示





Posize Playground https://www.pxcode.io/playground/posize

Posize 實作

- 實作
  - 主要以 一個 3 x 3 的 **CSS Grid** 為核心
  - 語法直接使用 CSS Grid 的語法
    - **ex. 2fr minmax(20px, 1fr) 3fr**

grid-template-columns: **1fr 3fr 2fr**

grid-template-rows: **1fr 2fr 1fr**

	top	
left	content	right
	bottom	

- React 版本使用 Functional Components
- HTML 版本使用 [Web Components]

`<px-posize> … </px-posize>`

```
<px-posize
  x="20px 546fr 558fr"
  y="20px 379px 191fr"
  md-x="1fr minmax(400px,2fr) 1fr"
  md-y="150px 379px 1fr"
  xs-x="10px 1fr 10px"
  xs-y="150px 379px 1fr">
  <div class="content">
    ...
  </div>
</px-posize>
```

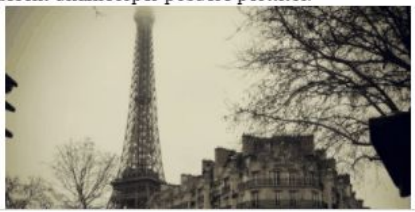
Why Posize？

- Posize 有什麼好處？ 用 CSS Margin 不是挺好的嗎？
- Posize
  - 目標在於「Fluid」，能依**「比例」**
  - CSS Grid 同時支援**「比例」與「百分比」(fr v.s. %)**
  - 直覺上，**「比例」**與**「百分比」**只是不同的表示法
  - 實質上
    - 百分比 → 各 child 各別參考 parent 的寬度
    - 比例 → 先尊重各 child 最小需要的寬度後，再以剩下的空間依比例分配

百分比(%) 與 比例 (fr) 比較演示

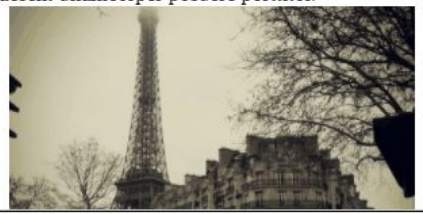
比例 2fr 5fr 3fr

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus feugiat posuere lectus, a rutrum est. Praesent ullamcorper posuere porttitor.



百分比 20% 50% 30%

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus feugiat posuere lectus, a rutrum est. Praesent ullamcorper posuere porttitor.



百分比 ⇒ 強制百分比，不會先尊重子內容

Posize v.s. Margin 總結比較

Posize [github]

- 優點
  - 完整 fluid 模式，符合 constraints 的行為
  - 同時支援百分比與比例 % + fr
  - 支援 minmax 與 % + fr 同時使用
  - 語法簡潔，MediaQuery 也是一目瞭然
  - 不會有 Margin Collapse 問題
- 缺點
  - 不支援負值 (margin 可以負值)
  - 需要 component 而不是一個

CSS Margin

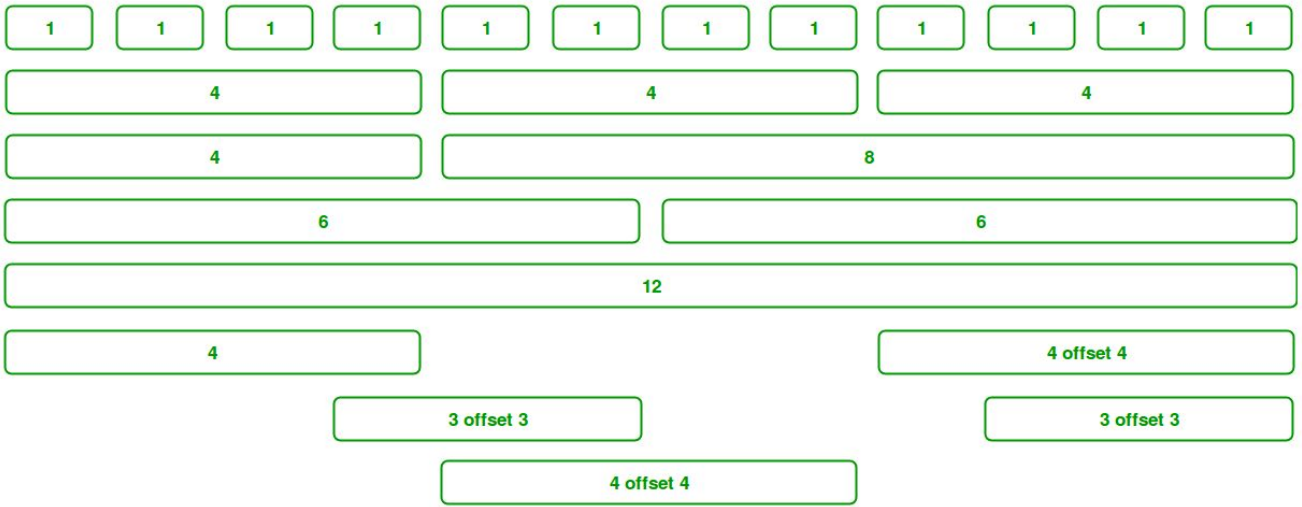
- 完全依照 parent 的百分比，不考慮動態內容的大小
- 做不到垂直依比例，因為是以 parent 的寬為參考
- 無法同時做到百分比與最大最小值
- ⇒ 多數的解決之道，大量使用 MediaQuery 來特解遇到的問題

```
<px-posize
  x="20px 546fr 558fr"
  y="20px 379px 191fr"
  md-x="1fr minmax(400px,2fr) 1fr"
  md-y="150px 379px 1fr"
  xs-x="10px 1fr 10px"
  xs-y="150px 379px 1fr">
  <div class="content">
    ...
  </div>
</px-posize>
```

Fusion Flex

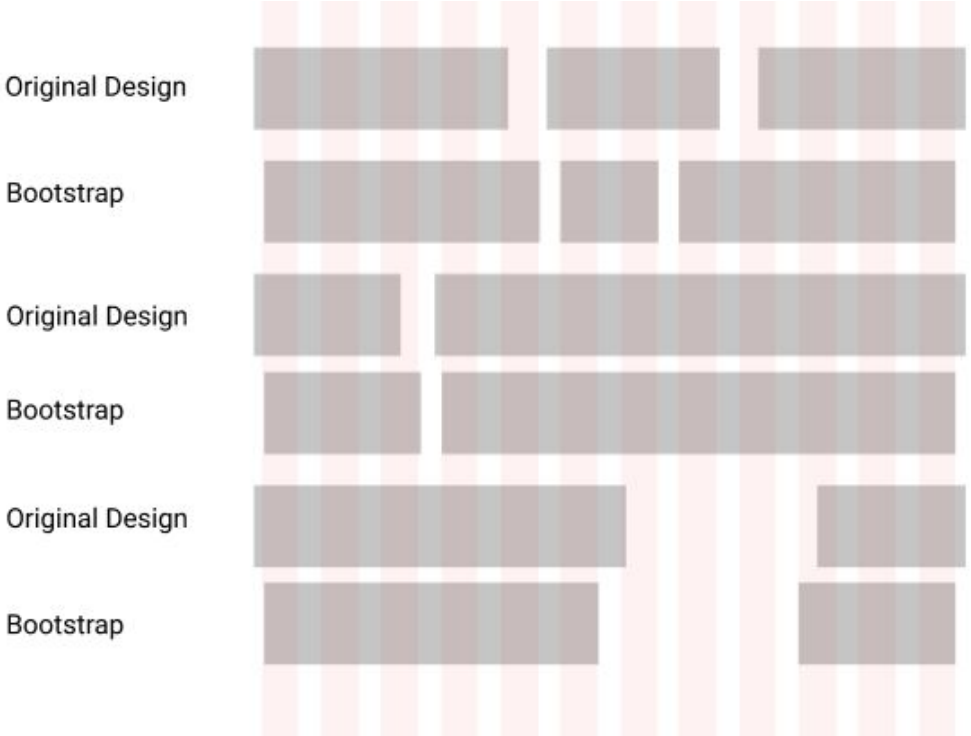
一開始 是想解決 **響應式排版** 的難題

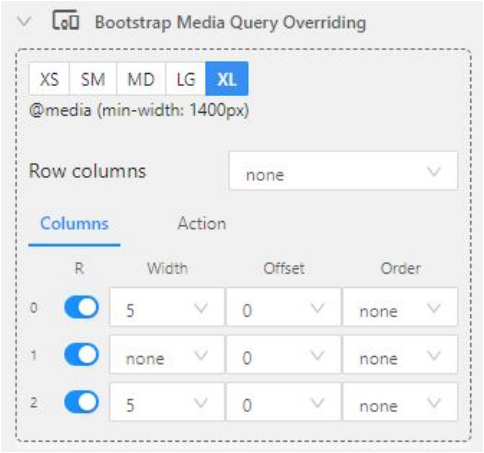
- bootstrap 提供了一個很好的解法
  - 核心利用 Grid System - 12 網格系統，同時 **排版**又好調 **RWD**
  - 但是設計檔不照著網格系統做， **bootstrap 就會跑版 !!!**



bootstrap in pxCode

- pxCode 也支援了 bootstrap，但痛苦的是會一套用會跑版
- 難道沒辦法 同時擁有 bootstrap 的彈性，又能夠不跑版嗎？

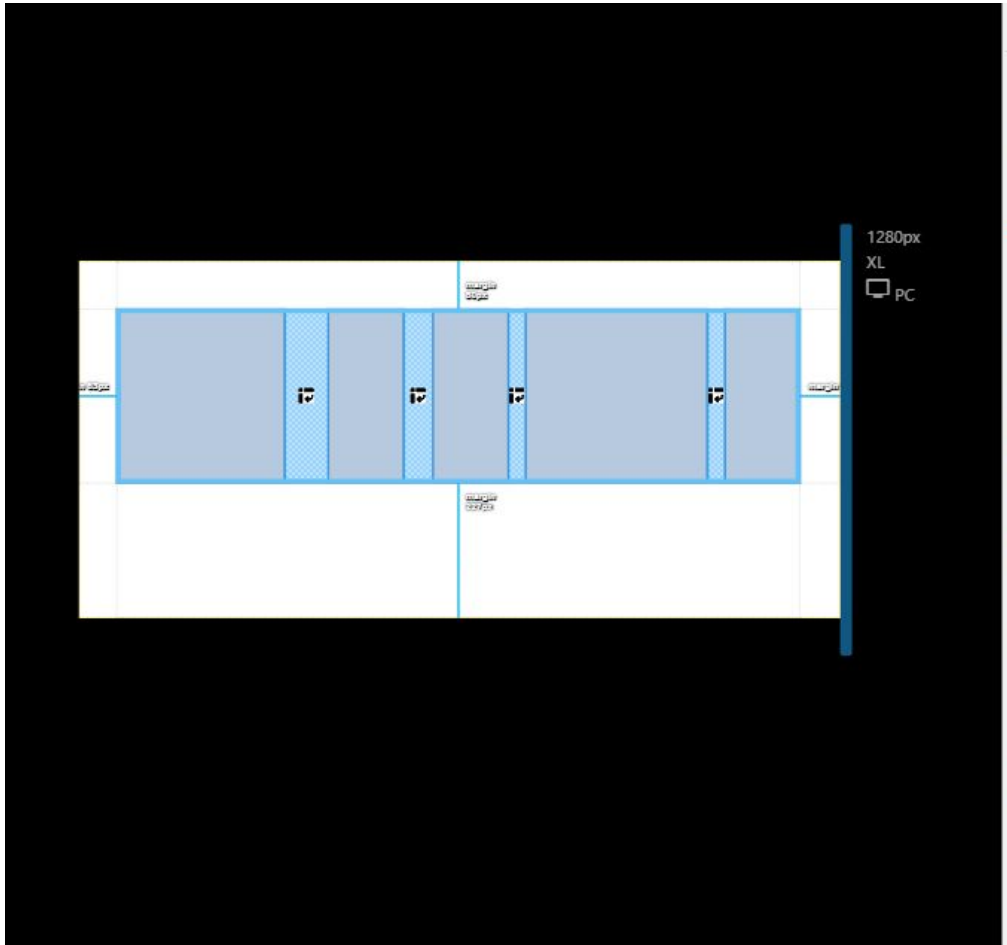




Fusion Flex

- 魚 與 熊掌 都要兼得 → 不跑版的 bootstrap
- 參考了 bootstrap 的實作概念
- Fusion Flex 概念
  - 全域 CSS, 僅使用 CSS Flex, 原先機制已十分彈性 - grow / shrink / basis
  - 要 Pixel Perfect 不跑版, 切出 spacer 空隔
  - 要 Grid System 的彈性, 搭配 MedaQuery
    - spacer 可切換成 break 換行點
    - 根本 break 切出多組後, 組內以 px 與 fr 作分配
  - Pure CSS without any dependencies 不仰賴任何 library

Fusion Flex 演示



Fusion Flex | Code

FLEXBOX

display: flex
flex-direction: row

Basic | More | Actions

	flex-grow		flex-shrink		flex-basis	(min-w)
0	Grow	1	Shrink	1	283 px	
1s	none	0	Shrink	1	73 px	
2	Grow	1	Shrink	1	127 px	
3s	none	0	Shrink	1	49 px	
4	Grow	1	Shrink	1	127 px	
5s	none	0	Shrink	1	29 px	
6	Grow	1	Shrink	1	306 px	
7s	none	0	Shrink	1	29 px	
8	Grow	1	Shrink	1	127 px	

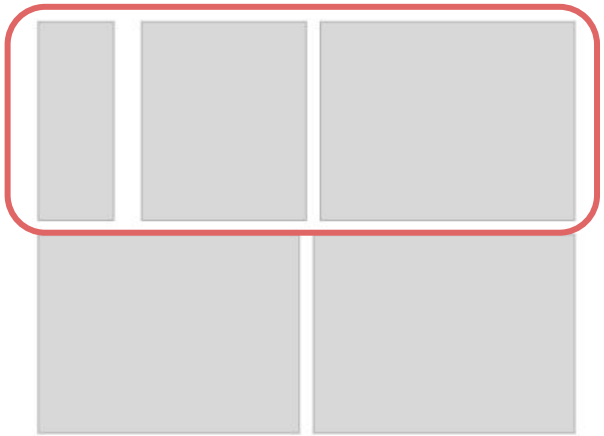
Wrap: nowrap

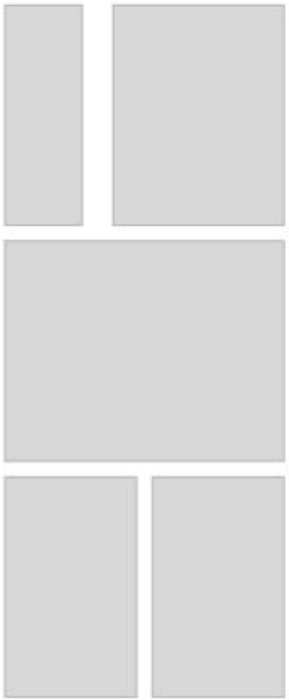
Justify Content: flex-start

RWD Flexbox Media Query Overriding

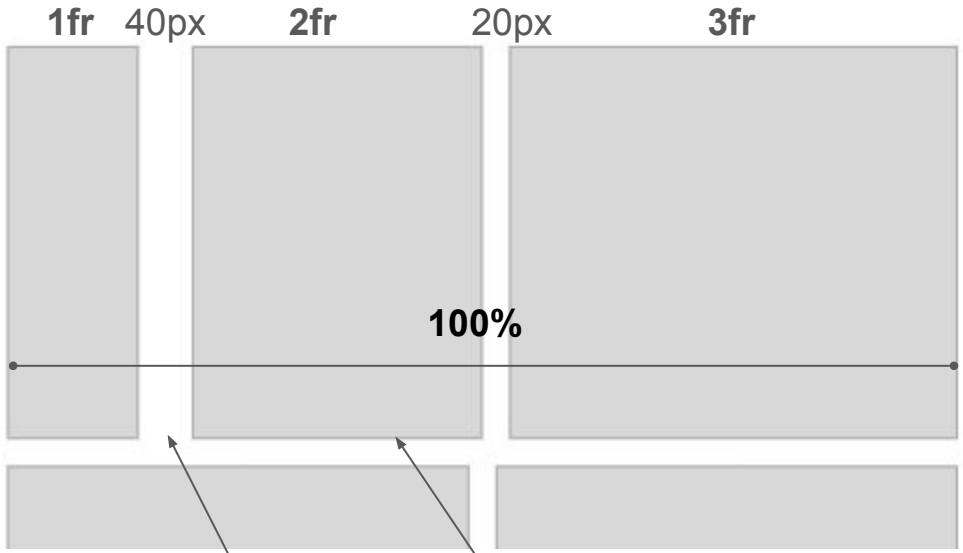
Fusion Flex 實作

- 參考 bootstrap
  - spacer 變成 break points => basis 100%
  - 分組後的分配, 例如是 **1fr 40px 2fr 20px 1fr**
    - 搭配 cal + basis % 達成





Fusion Flex Code (codepen)




```html
<div class="sample-flex layout">
  <div class="sample-flex-col"><div class="sample-rect layout"></div></div>
  <div class="sample-flex-spacer"></div>
  <div class="sample-flex-col1"><div class="sample-rect layout"></div></div>
  <div class="sample-flex-spacer1"></div>
  <div class="sample-flex-col2"><div class="sample-rect layout"></div></div>
  <div class="sample-flex-spacer2"></div>
  <div class="sample-flex-col3"><div class="sample-rect layout"></div></div>
  <div class="sample-flex-spacer3"></div>
  <div class="sample-flex-col4"><div class="sample-rect layout"></div></div>
</div>
```

計算方式

(100% - 40px - 20px) * 2 / 6

⇒ **33.33% - 20px**

```scss
.sample-flex-spacer {
  display: flex;
  flex: 0 1 73px;
  @include md {
    flex: 0 0 40px;
  }
}
```

```scss
.sample-flex-col1 {
  display: flex;
  flex: 1 1 127px;
  @include md {
    flex: 0 0 calc(33.3333333333333% - 20px);
  }
  @include xs {
    flex: 0 0 calc(66.66666666666666% - 20px);
  }
}
```

總結

- 前端開發 的 技術還是持續演化與推進
- 設計 與 前端開發 流程會持續進步
  - Handoff Process (ex. Zeplin) → Design Token → Design to Code
- Design To Code 問題目前還是十分新且困難的題目
  - pxcode 在這二年的努力，小小推進了幾步，解決切版痛點的長征還要繼續努力
- 歡迎大家試用，多多回饋跟建議，讓我們能改善產品

Thanks for Listening