# CGRectDivide(everything)

The procedural alternative to Autolayout

@manuelmaly

Auto Layout is the de facto standard, so why bother looking for something else?

# AutoLayout: Rant

- Difficult to debug
- Requires me to keep multiple constraints in mind at once
- Too magical to ever get a feeling of tight control over layout
- On a sidenote, Interface Builder burnt me more than once 😿

A solution:

Procedural layout

# Procedural Layout

- Lay out elements in code, by handcrafting CGRects and contentInsets
- Harness the power of DRY, methods, and constants
- Align vertically first, then horizontally
- Use either your own CGRect gen methods, or `CGRectDivide()`

# Procedural Layout Pros

- Blazingly fast - especially needed in UITableViewCells
- Puts less strain on your short term memory
- Easy to debug

# Procedural Layout Cons

- Hard to parse visually if not properly documented
- Often more code than AL
- No WYSIWYG

# Ok, so what does `CGRectDivide()` do?

```
void CGRectDivide (
    CGRect rect,
    CGRect *slice,
    CGRect *remainder,
    CGFloat amount,
    CGRectEdge edge
);
```

Cuts `rect` into two parts: `slice` and `remainder`. The cut follows the axis of `edge` (left/right/top/bottom edge). The distance of the cut to `edge` is defined by `amount`.

# TL;DR

`CGRectDivide()` cuts available space to pieces like scissors.

# Obj-C Demo

Repo at http://tiny.cc/cgrectdivide

# Swift

```
func rectsByDividing(
    atDistance: CGFloat,
    fromEdge: CGRectEdge
) -> (
    slice: CGRect,
    remainder: CGRect
)
```

Tuple goodness 😏

# Thanks for your patience!

@manuelmaly