orange

Overview

A **script** consists of one or more **bindings**. Bindings are written like this:

```
keypath <= expression
```

The result `expression` is assigned to the location specified by `keypath`.

A keypath can specify multiple objects. can be multiple objects. For example, in a `UIView` you could simultaneously move the left edge of all subviews to 10 pixels from the left edge of the view:

```
.subviews.left <= 10.0
```

You can nest keypaths. This allows you to set multiple properties of an object specified by a keypath at the same time, like this:

```
.subviews {
    .left <= 10.0;
    .right <= 100.0;
}
```

This sets both `left` and `right` for all subviews of a view.

Normally bindings are always ‘live’. Changes in the expression are applied at the destination keypath immediately. You can change this with a trigger. The only trigger currently supported is `layout`. Any bindings enclosed in a layout trigger are applied only after each invocation of `-layoutSubviews`.

```
<layout> {
    .subviews {
        .left <= 10.0;
        .right <= 100.0;
    }
}
```

When you compile your script, you can also pass in object references. This lets you refer to objects that cannot be specified via keypath in your script. You could center an arbitrary `UILabel` in a parent `UIView` like this:

```
[ OrangeScript  scriptWithContext:parentView
                           format:@"%@ { .position
      <= .superview.bounds.middle }", myLabel ] ;
```

Expressions

Expressions are written in prefix notation.

Examples:

addition

```
+ 5 10
```

subtraction (take 5 from 10)

```
- 10 5
```

ternary (if/then/else) operator:

```
#
# if a.hidden then result is 0
# otherwise result is a.width
#

? a.hidden 0 a.width
```

you can also optionally group subexpressions using parentheses:

```
#
# add 15 to (10 - 7)
#

+ ( - 10 7 ) 15
```

creating Size values:

```
CGSize width height
```