@isaacs/balanced-match

A hybrid CJS/ESM TypeScript fork of balanced-match.

Match balanced string pairs, like `{` and `}` or `<b>` and `</b>`. Supports regular expressions as well!

Example

Get the first matching pair of braces:

The matches are:

```
$ node example.js
{ start: 3, end: 14, pre: 'pre', body:
       'in{nested}', post: 'post' }
{ start: 3,
  end: 9,
  pre: 'pre',
  body: 'first',
  post: 'between{second}post' }
{ start: 3, end: 17, pre: 'pre', body:
       'in{nest}', post: 'post' }
```

API

const m = balanced(a, b, str)

For the first non-nested matching pair of `a` and `b` in `str`, return an object with those keys:

- **start** the index of the first match of `a`
- **end** the index of the matching `b`
- **pre** the preamble, `a` and `b` not included
- **body** the match, `a` and `b` not included
- **post** the postscript, `a` and `b` not included

If there's no match, `undefined` will be returned.

If the `str` contains more `a` than `b` / there are unmatched pairs, the first match that was closed will be used. For example, `{{a}` will match `['{', 'a', '']` and `{a}}` will match `['', 'a', '}']`.

const r = balanced.range(a, b, str)

For the first non-nested matching pair of `a` and `b` in `str`, return an array with indexes: `[ <a index>, <b index> ]`.

If there's no match, `undefined` will be returned.

If the `str` contains more `a` than `b` / there are unmatched pairs, the first match that was closed will be used. For example, `{{a}` will match `[ 1, 3 ]` and `{a}}` will match `[0, 2]`.