The theme is based on block layout. Or block grid, you can call it whatever you like. The fact is, it gives the page this smooth, measured look, unlike masonry grid, which might look a bit messy sometimes. To create these blocks you must follow a defined structure.

First of all, you need to use Materialize's grid (http://materializecss.com/grid.html) to define the width of your block. Usually you'd start with a row and use a div with `col s12 l6` class in order to make it full width on small devices and half width for medium and large screens.

Then include a div with a corresponding class (`rectangle`, `rectangle-vertical` or `square`), which will have an image stretch to its whole size (via `background-size: cover`) and finally a div containing your content. The whole thing would look like this:

```
<div class="row">
  <div class="col s12 l6">
    <div class="rectangle" style="background-image: url(path/to/image)">
      <div class="content">
        Your awesome content
      </div>
    </div>
  </div>
</div>
```

Extending the layout

You can extend those classes and create your own shapes, say a vertical rectangle. All you need to do is define it in your styles, like this:

```
.vertical-rectangle {
  padding-bottom: 200%; /* will create a rectangle with height = width * 2 */
}
```

There's no need to specify height of those shapes, since padding will do the trick and size them dynamically, depending on the width. Think of it as of a proportion: `padding-bottom: 100%` will give you a 1:1 square, `padding-bottom: 50%` will give you a 2:1 rectangle.

Improving imperfections

Sometimes it's hard to find the right proportions, if your item is being stretched across several columns and column paddings. That's when you need to add a little bit of Javascript in order to help measure the items height. A good example can be found in `work-3-columns` and `work-4-columns` pages. You define a class for the main item, say `.rectangle-8-of-12` and add another class `.double-col` for the column, the height of which we're gonna use to assign to the main item.

Then simple use Javascript to tell that the `.rectangle-8-of-12` should have the height that is equal to that of `.double-col` element, kinda like this:

```
function resizeRectangle () {
    var $rectangle = $('.rectangle-8-of-12');
    var $doubleCol = $('.double-col');

    if (window.innerWidth > 992) {
        $rectangle.css({
            height: $doubleCol.height(),
            paddingBottom: 0
        });
    } else {
        $rectangle.css({
            height: 0,
            paddingBottom: 'calc(120% + 1.5rem)'
        });
    }
}

resizeRectangle();

$(window).on('resize', resizeRectangle);
```

The `resizeRectangle` function is being run on page load as well as window resize. There is also a check for screen resolution and in case it's smaller than medium screen (991px), then the block gets a `padding-bottom` of `calc(120% + 1.5rem)`.

Textillate (http://textillate.js.org) hover effect

This effect can be used by using a `.tlt` class on the title, like this:

```
<h3 class="tlt">Colorful ice cream</h3>
```

Make sure you load the script via $.getScript (https://api.jquery.com/jquery.getscript/):

```
$.getScript('./assets/js/textillate.js')
    .done(function (script, status) {
        console.log('Textillate script status', status);
    })
    .fail(function (error) {
        console.error('Textillate script error', error);
    });
```

Feel free to play with the options in order to achieve a more personalised and custom effect. The default options we provide are:

```
$('.tlt').textillate({
    loop: false,
    minDisplayTime: 0,
    initialDelay: 0,
    autoStart: false,
    'in': {
        effect: 'fadeIn',
        delay: 25
    },
    out: {
        effect: 'fadeOut',
        delay: 25
    }
});
```