Migrating to Meteor 1.12

Most of the new features in Meteor 1.12 are either applied directly behind the scenes (in a backwards compatible manner) or are opt-in. For a complete breakdown of the changes, please refer to the changelog.

The above being said, there are some breaking changes to note.

Importing types

When importing types in Typescript, you might need to use the "type" qualifier, like so:

```
import { Point } from 'react-easy-crop/types';
```

to

```
import type { Point } from 'react-easy-crop/types';
```

Because now `emitDecoratorsMetadata` is enabled.

Typescript upgraded to 4.1.2

Refer to typescript breaking changes before migrating your existing project, from 3.7.6 to 4.1.2: https://github.com/Microsoft/TypeScript/wiki/Breaking-Changes

Migrating from a version older than 1.11?

If you're migrating from a version of Meteor older than Meteor 1.11, there may be important considerations not listed in this guide (which specifically covers 1.11 to 1.12). Please review the older migration guides for details:

- Migrating to Meteor 1.11 (from 1.10.2)
- Migrating to Meteor 1.10.2 (from 1.10)
- Migrating to Meteor 1.10 (from 1.9.3)
- Migrating to Meteor 1.9.3 (from 1.9)
- Migrating to Meteor 1.9 (from 1.8.3)
- Migrating to Meteor 1.8.3 (from 1.8.2)
- Migrating to Meteor 1.8.2 (from 1.8)
- Migrating to Meteor 1.8 (from 1.7)
- Migrating to Meteor 1.7 (from 1.6)
- Migrating to Meteor 1.6 (from 1.5)
- Migrating to Meteor 1.5 (from 1.4)

- Migrating to Meteor 1.4 (from 1.3)
- Migrating to Meteor 1.3 (from 1.2)