Migrating to Meteor 1.7

Most of the new features in Meteor 1.7 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 a few items that require additional attention, which are outlined below.

Babel / `meteor-node-stubs` updates

After updating to Meteor 1.7 or 1.7.0.1, you should update the `@babel/runtime` npm package (as well as other Babel-related packages), and the `meteor-node-stubs` package, to their latest versions.

```
meteor npm install @babel/runtime@latest meteor-node-stubs@latest
```

Mongo 3.6

Mongo has been upgraded to version 3.6.4 for 64-bit systems, and 3.2.19 for 32-bit systems.

After upgrading an application to use Mongo 3.6.4, it has been observed that attempting to run that application with an older version of Meteor (via `meteor --release X`), that uses an older version of Mongo, can prevent the application from starting. This can be fixed by either running `meteor reset` (**WARNING:** will wipe your local DB), or by repairing the Mongo database. To repair the database, find the mongod binary on your system that lines up with the Meteor release you are jumping back to, and run `mongodb --dbpath your-apps-db --repair`. For example:

```
~/.meteor/packages/meteor-tool/1.6.0_1/mt-os.osx.x86_64/dev_bundle/mongodb/bin/mongod --dbpa
```

Migrating from a version older than 1.6?

If you're migrating from a version of Meteor older than Meteor 1.6, there may be important considerations not listed in this guide (which specifically covers 1.6 to 1.7). Please review the older migration guides for details:

- 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)