Upgrade Guide

0.18.x -> 0.19.0

HTTPS Proxies Routing through an https proxy now requires setting the `protocol` attribute of the proxy configuration to `https`

0.15.x -> 0.16.0

`Promise` Type Declarations The `Promise` type declarations have been removed from the axios typings in favor of the built-in type declarations. If you use axios in a TypeScript project that targets `ES5`, please make sure to include the `es2015.promise` lib. Please see this post for details.

0.13.x -> 0.14.0

TypeScript Definitions The axios TypeScript definitions have been updated to match the axios API and use the ES2015 module syntax.

Please use the following `import` statement to import axios in TypeScript:

```
import axios from 'axios';

axios.get('/foo')
  .then(response => console.log(response))
  .catch(error => console.log(error));
```

`agent` Config Option The `agent` config option has been replaced with two new options: `httpAgent` and `httpsAgent`. Please use them instead.

```
{
  // Define a custom agent for HTTP
  httpAgent: new http.Agent({ keepAlive: true }),
  // Define a custom agent for HTTPS
  httpsAgent: new https.Agent({ keepAlive: true })
}
```

`progress` Config Option The `progress` config option has been replaced with the `onUploadProgress` and `onDownloadProgress` options.

```
{
  // Define a handler for upload progress events
  onUploadProgress: function (progressEvent) {
    // ...
  },

  // Define a handler for download progress events
  onDownloadProgress: function (progressEvent) {
    // ...
  }
}
```

0.12.x -> 0.13.0

The `0.13.0` release contains several changes to custom adapters and error handling.

Error Handling Previous to this release an error could either be a server response with bad status code or an actual `Error`. With this release Promise will always reject with an `Error`. In the case that a response was received, the `Error` will also include the response.

```
axios.get('/user/12345')
  .catch((error) => {
    console.log(error.message);
    console.log(error.code); // Not always specified
    console.log(error.config); // The config that was used to make the request
    console.log(error.response); // Only available if response was received from the server
  });
```

Request Adapters This release changes a few things about how request adapters work. Please take note if you are using your own custom adapter.

1. Response transformer is now called outside of adapter.
2. Request adapter returns a `Promise`.

This means that you no longer need to invoke `transformData` on response data. You will also no longer receive `resolve` and `reject` as arguments in your adapter.

Previous code:

```
function myAdapter(resolve, reject, config) {
  var response = {
    data: transformData(
      responseData,
      responseHeaders,
      config.transformResponse
    ),
    status: request.status,
    statusText: request.statusText,
    headers: responseHeaders
  };
  settle(resolve, reject, response);
}
```

New code:

```
function myAdapter(config) {
  return new Promise(function (resolve, reject) {
    var response = {
      data: responseData,
      status: request.status,
      statusText: request.statusText,
      headers: responseHeaders
    };
    settle(resolve, reject, response);
  });
}
```

See the related commits for more details: - Response transformers - Request adapter Promise

0.5.x -> 0.6.0

The `0.6.0` release contains mostly bug fixes, but there are a couple things to be aware of when upgrading.

ES6 Promise Polyfill Up until the `0.6.0` release ES6 `Promise` was being polyfilled using es6-promise. With this release, the polyfill has been removed, and you will need to supply it yourself if your environment needs it.

```
require('es6-promise').polyfill();
var axios = require('axios');
```

This will polyfill the global environment, and only needs to be done once.

`axios.success`/`axios.error` The `success`, and `error` aliases were deprecated in 0.4.0. As of this release they have been removed entirely. Instead please use `axios.then`, and `axios.catch` respectively.

```
axios.get('some/url')
  .then(function (res) {
    /* ... */
  })
  .catch(function (err) {
    /* ... */
  });
```

UMD Previous versions of axios shipped with an AMD, CommonJS, and Global build. This has all been rolled into a single UMD build.

```
// AMD
require(['bower_components/axios/dist/axios'], function (axios) {
  /* ... */
});
```

```
// CommonJS
var axios = require('axios/dist/axios');
```