There are two types of user tests: npm install and submodule.

NPM Install Tests

npm install tests have two purposes.

1. Test the user experience of a Typescript package.
2. Test compilation of a Javascript package without having to use submodules.

These tests have at least a package.json and a tsconfig.json.

In the Typescript-user case, there is also an index.ts that matches how the user is expected to use the library. However, today, most of these files are just `import x = require('x')`. This just makes sure that the typings exported by the project continue to work with new versions of Typescript, even if the authors aren't paying attention to new versions.

In the Javascript-source case, the tsconfig.json points into the node_modules directory: `include: ['node_modules/x/src']` or something similar. Many Javascript projects ship their source or something close to it, and an npm install is **much** easier than a git submodule.

Submodule Tests

Submodule tests let us test project source without needing to check in a copy to a repo that we own. This is better than our internal real-world code tests, because reproing errors is much easier: just clone the repo and run tsc. However, it requires an open source project that can build with a single call to tsc. That excludes modern Angular and many Microsoft-internal projects.

Also you have to use submodules.

Pitfalls of submodules

1. They are submodules!
2. The submodule repo name needs to be the same as the containing directory name.