JavaScript FAQ

JavaScript

Main articles: Client-Side JavaScript, Legacy JavaScript

Contents

Why do I get a "Tapestry is undefined" error on form submit? (5.3 and earlier)

This client-side error is clear but can be awkward to solve. It means your browser has not been able to load the tapestry.js file properly. The question is, why? It can be due to multiple reasons, some of them below:

- First, check if 'tapestry.js' is present in the head part of your resulting HTML page.
- If you have set the tapestry.combine-scripts configuration symbol to true, Tapestry generates one single URL to retrieve all the JS files. Sometimes, this can produce long URLs that browsers are unable to retrieve. Try setting the symbol to false.

> ⚠ This only applies to Tapestry 5.1.

- If you have included jQuery in conjunction with Tapestry's prototype, that will cause a conflict with the '$' selector used by both. In this case, you should put jQuery on top of the stack and turn on the jQuery.noConflict mode.
- Also, if you have included a custom or third-party JS library on top of the stack that causes the JavaScript parsing to fail, then check the JavaScript syntax in that library.
- If you have used a tool to minimize your JavaScript libraries, this can lead to JavaScript syntax errors, so check if it works with all the JavaScript files unpacked.

What's the difference between the `T5` object and the `Tapestry` object in the browser? (5.3 and earlier)

Both of these objects are *namespaces*: containers of functions, constants, and nested namespaces.

The `T5` object is a replacement for the `Tapestry` object, starting in release 5.3. Increasingly, functions defined by the `Tapestry` object are being replaced with similar or equivalent functions in the `T5` object.

This is part of an overall goal, spanning at least two releases of Tapestry, to make Tapestry JavaScript framework agnostic; which is to say, not depend specifically on Prototype or jQuery. Much of the code in the `Tapestry` object is specifically linked to Prototype and Scriptaculous.

The `T5` object represents a stable, documented, set of APIs that are preferred when building components for maximum portability between underlying JavaScript frameworks. In other words, when building component libraries, coding to the `T5` object ensures that your component will be useful regardless of whether the final application is built using Prototype, jQuery or something else.