Extra - Proposal for ES7 Types

Friday, September 12, 2014 2:33 PM

Hi <Recipients>,

Jonathan Turner (of TypeScript fame) and I are hoping to present about our static types proposal at the next TC39. In advance of that I was hoping to get some early feedback from you on the proposal. We know that there are a large number of developers who get a lot of value from the static, erasable types provided by TypeScript, and we think this would be a good avenue to pursue for standardizing types in ECMAScript itself. Having these types available in ECMAScript would certainly be a boon to existing TypeScript users, and would also enable a wide variety of tools to begin using these annotations to provide early errors and generally improve JS tooling, and provide a standard type definition format for library authors.

We're looking to put a proposal forward for both types and type checking semantics in ES7. The former can stand alone as something similar to what Guido proposed as part of Python 3.5: a syntax-only spec change that allows type annotations without enforced semantics. While we think TypeScript provides great type checking semantics, we are amenable to leaving this unspecified in order to allow for innovation in this space. Any thoughts as to whether you would prefer purely syntactic additions or also standardizing type checking semantics would be appreciated.

As for the details of the proposal, as mentioned we have two layers. Layer 0 is just the type annotation syntax. The syntax includes:

- optional annotations for function parameters and return types, variables, and classes.
- Type casting of arbitrary expressions
- Interface declarations that allow structural typing of JavaScript objects (including functions, arrays, etc).

TypeScript does not yet have a complete answer on what syntax is used for typing modules, symbols, and some other ES6 additions so there will be other areas of discussion in this space. That said, the TypeScript team is actively working now to align with ES6 and we will likely incorporate these updates into our proposal.

Layer 1 uses the type annotations added by layer 0 to allows the developer to opt in to additional early errors based on the standardized type checking semantics. Our proposal for Layer 1 is effectively to adopt TypeScript's type checking semantics. These semantics have seen broad usage across millions of lines of real world code. Again, a similar caveat as above about new ES6 features apply.

More details will of course be forthcoming, but in the meantime we wanted to get your thoughts on this idea before moving these proposals to "stage 0". Please share any thoughts you have!

Thanks,
Brian