entwickler.press
shortcuts

Next Level JavaScript

Schlagworte

Jakob Westhoff, Michael Wager,
Stefanos Aslanidis, Robert Rieger,
Peter Kern, Christian Ringler

Jakob Westhoff, Michael Wager, Stefanos Aslanidis,
Robert Rieger, Peter Kern, Christian Ringler

Next Level JavaScript –

Schlagworte

ISBN: 978-3-86802-497-5

Ein Imprint der Software & Support Media GmbH

1 JavaScript als Plattform

JavaScript besitzt in der Webentwicklung einen hohen Stellenwert. Doch nicht nur dort fühlt sich diese Sprache zu Hause. Mit Node.js ist sie ebenso auf Servern vertreten, und IDEs erlauben mit ihr die Erstellung eigener Module. Auch auf Fernsehern, Spielekonsolen und Embedded-Systemen ist sie anzutreffen. Kurzum: Die Verbreitung von JavaScript scheint kaum mehr aufzuhalten. Doch was bedeutet das konkret für die Entwicklung der Sprache? Ist JavaScript mittlerweile vielleicht mehr geworden, als eine reine Programmiersprache?

JavaScripts mannigfaltige Sprachfeatures lassen extrem elegante und mächtige Problemlösungen entstehen, führen jedoch gleichzeitig häufig zu unleserlichem und extrem schwer wartbarem Code. Viele sind der Meinung, dies sei ein sprachinhärentes Problem, das sich mit JavaScript nicht lösen lasse. Die Konsequenz: Eine alternative Sprache muss her! Doch wie soll das funktionieren? Browser, die zu den Haupteinsatzgebieten von JavaScript zählen, können mit keiner anderen Sprache umgehen. Natürlich könnten neue Sprachen erfunden werden. Doch aus Erfahrung vergehen etliche Jahre, bis sich diese soweit etabliert haben, dass sie produktiven Einsatz finden – falls dies überhaupt gelingt.

Die Entstehung von Transpilern

Eine einfache, wenn auch geniale Lösung des Ein-Sprachen-Problems ist die Verwendung sog. Transpiler, auch Transcompiler oder Source-to-Source-Compiler genannt. Transpiler sind Compiler, die eine Eingabesprache nicht in Bytecode für eine spezielle Plattform oder VM übersetzen, sondern in eine andere Hochsprache überführen. In Zeiten von interpretierten und mit Just-in-Time-Compilern optimierten Sprachen kann diese Technik eine Möglichkeit bedeuten,

alternative Sprachen auch in scheinbar isolierten Systemen wie JavaScript zum Einsatz zu bringen.

CoffeeScript – Ein bekannter Transpiler

Ein Vertreter der Sprachen, für die ein Transpiler existiert, ist CoffeeScript [1]. Wenig überraschend ist die Zielsprache dieses Compilers JavaScript. Applikationen können demnach in CoffeeScript geschrieben und anschließend in JavaScript übersetzt werden. Hierbei muss die Ausgangssprache keine Ähnlichkeit mit JavaScript besitzen.

Eine Klassendefinition in CoffeeScript demonstriert recht eindrucksvoll, dass die Ausgangssprache nicht zwangsläufig Ähnlichkeit mit der Zielsprache besitzen muss (Listing 1.1).

```
class Shape
draw: ->
throw new Error("Abstract method.")

class Circle extends Shape
draw: ->
# Draw the Circle
```

Listing 1.1

CoffeeScript besitzt Klassen, JavaScript lediglich Prototypen. CoffeeScript verwendet Einrückungen zur Identifikation von Blöcken, JavaScript geschweifte Klammern. Funktionen werden ohne das Keyword function definiert. Es handelt sich also tatsächlich um eine neue Sprache, die sich gravierend von der Zielsprache unterscheidet. Trotzdem ist es möglich, das vorgestellte Programm mithilfe des CoffeeScript-Transpilers in JavaScript-Code zu übersetzen (Listing 1.1).

TypeScript – Microsofts Idee von JavaScript

CoffeeScript ist bei Weitem nicht die einzige neue Sprache dieser Gattung. Mit TypeScript [2] hat Microsoft vor einiger Zeit einen weiteren Vertreter der JavaScript-Alternativen ins

Rennen geschickt. Im Vergleich zu Sprachen wie CoffeeScript ist TypeScript keine komplett neue Sprache, sondern lediglich ein Superset, also eine Erweiterung von JavaScript.

Jeglicher gültige JavaScript-Code ist demnach auch gültiger TypeScript-Code. Allerdings besitzt TypeScript diverse zusätzliche Features, die JavaScript aktuell nicht bietet. So z. B. ein Modulsystem, Klassen, Interfaces und – wie der Name der Sprache bereits vermuten lässt – ein statisches Typensystem. Listing 2 zeigt zwei primitive TypeScript-Klassen und ein Interface. Diese wiederum werden verwendet, um einen Kreis in einen Canvas zu zeichnen. Das Beispiel demonstriert anschaulich die Verwendung von statischen Typen.

Auch TypeScript ist ein Transpiler, der seine erweiterten Sprachfeatures zurück in JavaScript übersetzt, um diese in aktuellen Engines zur Ausführung zu bringen. Der Compiler übersetzt in diesem speziellen Fall nicht nur den Quelltext, sondern überprüft und erzwingt darüber hinaus auch die Verwendung korrekter Typen.

Weitere Transpiler

Neben den beiden kurz vorgestellten Sprachen CoffeeScript und TypeScript gibt es eine schier unendliche Menge weiterer Transpiler. Die folgende Liste enthält einige der bekannteren Vertreter: LispyScript, Moescript, Dart, JavaScript++, StratifiedJS, Objective-J. Eine weitaus vollständigere Liste von existierenden Transpilern für JavaScript findet sich z. B. im CoffeeScript-Wiki [3].

Emscripten und Co. – Der nächste logische Schritt

Die bisherige Betrachtung galt vornehmlich Sprachen, die allesamt eines gemeinsam hatten: Sie wurden speziell zu dem Zweck entworfen, in JavaScript übersetzt zu werden. Was ist jedoch mit bereits existierenden Sprachen wie C, C++, C# oder Python? Lassen sich diese vielleicht auch in JavaScript-