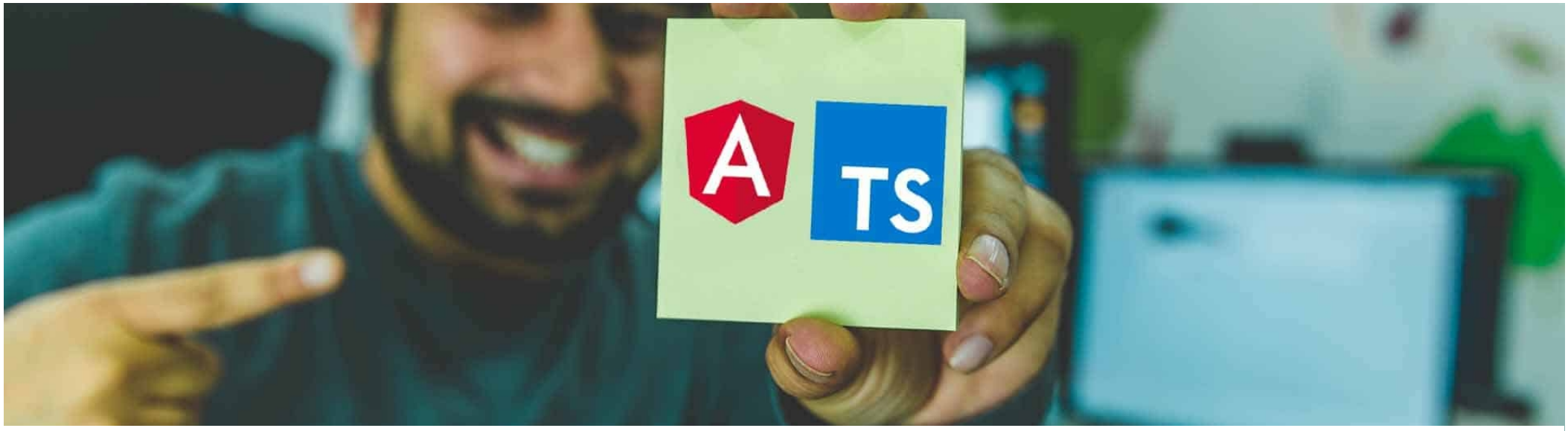

Inhaltsverzeichnis

Konsistenz
Produktivität
Wartbarkeit
Modularität
Frühzeitige Fehlererkennung
Zusammenfassung

In den letzten Monaten wurde mir in Schulungen immer wieder dieselbe Frage zu Angular gestellt. Bei Gesprächen mit Unternehmensführern über die Richtung, in die Web-Technologien gehen, bei Kunden, denen ich in Bezug auf ihre Architektur behilflich war. Immer wieder ein und dieselbe Frage: „Was sind die wichtigsten Vorteile, die Angular und TypeScript unseren Entwicklungsteams bieten können?“

Es ist eine großartige Frage, die gestellt werden sollte, bevor Sie in neue Technologien oder Frameworks einsteigen. Die meisten der Fragesteller sind Technologie-Manager oder -Direktoren, die daran interessiert sind, die Vorteile zu verstehen, die Angular ihren Teams aus technischer und auch aus nicht-technischer Sicht bieten kann. Sie sind besorgt über die Aufwände zur Wartung von Anwendung, die Produktivität der Entwickler, die Bearbeitung von Änderungswünschen, die Lebensdauer des Frameworks, das Tempo der Technologie und vieles mehr.

Nachdem ich diese Frage immer und immer wieder gehört habe, ist es nun an der Zeit, meine 5 fünf wichtigsten Gründe für die Verwendung von Angular und TypeScript zu formulieren. Es gibt sicherlich VIELE andere, die ich auflisten könnte, aber hier sind meine Top-5-Vorteile:

- Konsistenz
- Produktivität
- Wartbarkeit
- Modularität
- Frühzeitige Fehlererkennung

Bevor ich fortfahre, möchte ich erwähnen, dass ich zwar mit Angular viel arbeite (und es genieße), aber ich bin kein Typ der nur auf ein Framework setzt. Warum? Wenn es eines gibt, was ich im Leben gelernt habe, dann „one size never fits all“. Wenn jemand vorschlägt, dass eine Technologie oder ein Framework * immer * die beste Option ist, sage ich gerne: „Gehen Sie in ein Schuhgeschäft und sagen Sie, dass Sie ein Paar Schuhe anprobieren möchten. Wenn Sie gefragt werden, welche Größe Sie möchten, antworten Sie einfach, dass es keine Rolle spielt und schauen, was dann passiert. “One size fits all“ passt nicht beim Kaufen von Schuhen und es passt auch nicht in der Technologie.

Mein Unternehmen (Wahlin Consulting) arbeitet mit vielen großen und kleinen Unternehmen zusammen. Einige verwenden AngularJS 1.x, einige sind auf Angular v2+ umgestiegen, einige verwenden React, andere erkunden Vue.js (und andere) und einige verwenden immer noch jQuery. Dieser Beitrag richtet sich an diejenigen, die sich mit Angular und TypeScript beschäftigen. Wenn Sie nicht an Angular / TypeScript interessiert sind, ist dieser Beitrag wahrscheinlich nichts für Sie.

Beginnen wir mit Vorteil Nr. 1.

Konsistenz

Die Codekonsistenz ist ein wichtiges Ziel, das bei der Implementierung von Code immer angestrebt werden muss. Wenn Sie oder Ihr Team Produktionsanwendungen unterstützen müssen, wissen Sie, wie wichtig Konsistenz ist und warum dies zu einer besseren Wartung führt. Was bietet Angular im Sinne von Konsistenz? Das gesamte Framework basiert auf Komponenten und Diensten, die Sie sich als Lego-Blöcke vorstellen können. Alle Komponenten und Dienste beginnen auf dieselbe Weise. Bspw. führen alle Angular-Komponenten folgendes aus:

- Den Import der erforderlichen ES2015-Module
- Die Definition eines @Component-Dekorators (Metadaten)
- Die Platzierung von Code in einer Komponentenklasse

Wie das funktioniert, zeigt das nachfolgende Bild:

Dieser Struktur ist immer identisch, unabhängig davon, welche Komponente Sie schreiben. Sicherlich gibt es Aspekte, die Sie hinzufügen können (implementieren Sie eine Schnittstelle wie OnInit oder entsprechende Alternativen, wenn Sie TypeScript verwenden, legen Sie Vorlagen inline oder in eine separate Datei ab, usw.), aber die Gesamtstruktur einer Komponente sieht immer gleich aus. Und das ist auch gut so, denn es sorgt von Anfang an für Konsistenz, wenn Teammitglieder mit dem Erstellen von Komponenten beginnen.

Ein weiterer großer Bereich der Konsistenz in Angular betrifft Services. In AngularJS 1.x können Sie zwischen Factories, Services, Providern, Values und Constants wählen, wenn Sie über Code nutzen, der in einer Anwendung wiederverwendet werden muss. Einige Entwickler bevorzugen Factories, während andere sich auf Services konzentrieren. Sie machen beide dasselbe, aber welches ist das „Richtige“? Die Antwort lautet: es ist ziemlich subjektiv. Wenn ein Team nicht einen Code Style für Teammitglieder vereinbart, macht jeder Entwickler sein eigenes Ding. Ich bezeichne ein solches Vorgehen gerne als „Cowboy-Codierung“ – natürlich möchte ich damit keine Cowboys damit beleidigen … :-)).

Glücklicherweise macht Angular die Entscheidung einfach, wie wiederverwendbarer Code in eine Anwendung eingefügt werden soll. Alles in diesem Szenario ist einfach eine Serviceklasse:

```
import { Injectable } from '@angular/core';
import { MyDependency } from './mydependency.service';

@Injectable()
class MyService {
```

```
    constructor(private myDependency: MyDependency) {}
}
```

Alle Abhängigkeiten, die der Service erfordert, können in seinen Konstruktor „eingefügt“ werden, wie im obigen Codebeispiel gezeigt (für die Abhängigkeit muss ein Provider registriert sein). Dies ist ein schönes Beispiel für die Konsistenz: Während Komponenten, Services und andere Typen sicherlich Instanzen der benötigten Objekte erstellen können, bietet Angular eine built-in-dependency, die die Objekte zur Laufzeit einfügt. Dies sorgt nicht nur für Konsistenz in einer Anwendung, sondern ermöglicht auch das situative Überschreiben von eingefügten Objekten, was in vielen Szenarien hilfreich sein kann.

Die Angular-Dokumentation enthält auch einen Style-Guide, den Teams als Ausgangspunkt verwenden können, um die Konsistenz projektweit zu verbessern. Wäre ich Direktor, Manager, Teamleiter oder einfach für die Gewährleistung der Konsistenz eines Teams zuständig, würde ich die erforderliche Zeit investieren, um einen teamspezifischen Style-Guide für jedes verwendete Framework zu erstellen.

Schließlich bietet Angular ein CLI-Tool, mit dem Sie initiale Projekte erstellen, verschiedene Funktionen zu einer Anwendung hinzufügen können (Komponenten, Services usw.), Tests durchführen, Builds ausführen, Lint-Code erzeugen und vieles mehr. Dies ist eine großartige Grundlage für Teams, um die Konsistenz zwischen Teammitgliedern und sogar mehreren Teams in einem Unternehmen zu verbessern.

Mein Fazit lautet daher: Die Konsistenz von Angular-Komponenten, Services, Pipes etc. hilft Teams, und sorgt für ein Gefühl mit dem Strom und nicht gegen ihn zu schwimmen. Und das führt direkt zum nächsten Vorteil: der Produktivität.

Produktivität

Mit der Konsistenz rückt auch die Produktivität ins Blickfeld. Grundsätzlich müssen sich Entwickler nicht so viele Gedanken machen, ob sie Dinge auf die „richtige Weise“ tun. Im wesentlichen sehen Komponenten und Services gleich aus, wiederverwendbarer Anwendungscode wird in Serviceklassen abgelegt, ES6 / ES2015-Module organisieren zugehörige Funktionen und ermöglichen, dass der Code self-contained und self-responsible ist. Daten werden mithilfe von Eingabeeigenschaften an Komponenten übergeben und lassen sich durch Verwendung von Ausgabeeigenschaften weitergeben.

Mit steigender Konsistenz erhalten Sie somit den zusätzlichen Nutzen der Produktivität. Sobald Sie verstanden haben, wie Sie eine Komponente schreiben, können Sie eine andere nach den gleichen allgemeinen Richtlinien und Codestrukturen schreiben. Sobald Sie verstanden haben, wie Sie eine Serviceklasse erstellen, können Sie leicht eine weitere erstellen. Sie werden schneller und besser, und wenn Sie all dies mit der Angular-CLI kombinieren, sind Sie konsistent und gleichzeitig produktiv.

Wenn Sie TypeScript zum Erstellen Ihrer Angular-Anwendungen verwenden, profitieren Sie außerdem von mehreren Produktivitätsvorteilen. In Editoren wie VS Code und WebStorm haben Sie während der Eingabe Zugriff auf eine robuste Code-Hilfe (Intellisense), die das Erkennen von Typen und der von ihnen angebotenen Funktionen erleichtert. Wenn Sie TypeScript-Schnittstellen verwenden, erhalten Sie sogar Codehilfe für die JSON-Daten, die von Aufrufen eines Back-End-Services zurückgegeben werden. Dies ist äußerst hilfreich, wenn verschiedene Daten- / Modellobjekte von Entwicklern verwendet und bearbeitet werden. TypeScript funktioniert natürlich nicht nur mit Angular (Sie können es mit React, AngularJS, Vue.js, Node.js und allen anderen JavaScript-Bibliotheken / Frameworks verwenden), aber es lässt sich gut in Angular integrieren.

Wartbarkeit

Ich liebe Open-Source-Projekte, kann aber auch verstehen, dass sich Personen oder Gruppen, die ein solches Projekt leiten, mit anderen Aspekten des Lebens auseinandersetzen müssen und manche Projekte somit nicht mehr aufrechterhalten werden können. Wenn Sie schon lange mit Open-Source-Software-Projekten gearbeitet haben, kennen Sie vielleicht entsprechende Hintergründe und Geschichten. Immer wenn ich ein Projekt für mich in Betracht ziehe, schaue ich mir die Anzahl der Mitwirkenden an, schaue wann es zuletzt aktualisiert wurde, und überprüfe die Issues an, um zu erkennen, wie regelmäßig sie behandelt werden. Ich bin sehr darum bemüht, nur Projekte, Module, Bibliotheken usw. zu verwenden und zu empfehlen, die

aktiv unterstützt werden, so dass meine Projekte und die Projekte meiner Kunden einfacher zu warten und auf dem neuesten Stand sind.

Ein engagiertes Team bei Google Building Angular in Kombination mit den Open-Source-Beiträgen der Community ist für mich persönlich ein RIESIGES Verkaufsargument. In der heutigen JavaScript-Welt von „Flavour of the Day“ weiß man natürlich nie, was morgen kommen wird, daher gibt mir das solide Fundament, das das Framework unterstützt, viel Vertrauen. Ein weiterer Bonus ist die Tatsache, dass Google Angular innerhalb des eigenen Unternehmens bei der Entwicklung von Anwendungen nutzt. Vielleicht werden Sie jetzt denken, dass dasselbe auch für React gilt… .und damit liegen Sie richtig. Allerdings möchte ich mich hier auf Angular konzentrieren.

Möglicherweise denken Sie auch: „Aber Dan – der Sprung von AngularJS 1.x auf Angular 2+ war riesig und definitiv nicht gut für Wartbarkeit unserer App!“ Ja, das ist ein valider Punkt – selbst mit Upgrade-Optionen. Der Sprung zwischen AngularJS und Angular war ein direktes Ergebnis der JavaScript-Sprache, die mit der ES6 / ES2015-Funktionalität enorme Fortschritte erzielte, kombiniert mit neuen Funktionen, die moderne Browser jetzt unterstützen können. Hätte das Angular-Team NICHT diesen Sprung vollzogen, würden wir Angular in kürzester Zeit als „Caveman Framework“ bezeichnen, da es nicht die neuesten und besten Funktionen nutzen würde, die die Leistung, Konsistenz, Produktivität, Wartbarkeit und Gesamtentwicklung unterstützen . Ich war froh als das Angular-Team diesen Schritt wagte. Natürlich habe auch ich keine Glaskugel, um in die Zukunft zu schauen (wenn jemand eine hat, würde ich sie gerne ausleihen), aber ich weiß, dass das Angular-Team sehr genau weiß, wie sich Änderungen des Frameworks auf Unternehmensprojekte auswirken (zumal Angular auch bei Google selbst häufig verwendet wird). Basierend auf dem, was ich gehört habe, bin ich überzeugt, dass die Veröffentlichung neuer Versionen zukünftig weniger Auswirkungen haben werden.

Neben der soliden Unterstützung, die Angular durch dem Angular-Team hinter den Kulissen erfährt, führt aber auch die beschriebene Konsistenz im Code zu einer einfacheren Wartbarkeit. In meiner Karriere durfte ich viele Jahre im Support arbeiten (ich hatte sogar viele Jahre lang den guten alten Pager im Einsatz), daher ist mir bewusst, wie wichtig konsistenter und einfach zu pflegender Code ist. Natürlich kann ein Team jedes Framework mit dem richtigen Style-Guide, Training und Wissen für eine einheitliche Anwendungsentwicklung nutzen. Das gilt nicht nur für Angular, sondern auch für viele andere Frameworks. Angular bietet jedoch einen sehr klaren Weg für das Schreiben von Code, was letztendlich zu einer vereinfachten Wartung führt. Und wenn Ansprechpartner in den Urlaub fahren (oder den Job wechseln), können leicht auch Kollegen gefundene Fehler beheben oder Änderungsanforderungen ohne weiteres bearbeiten.

Angular Code kann mit TypeScript erstellt werden (meiner Präferenz), was eine Vielzahl von Vorteilen für Unternehmen bietet. Im Kapitel „Frühzeitige Fehlererkennung“ werden ich nochmals auf Typescript und einige deutliche Wartungsvorteile eingehen.

Unabhängig davon, ob Ihr Team selbst den Support für eine Anwendung durchführt oder dies durch ein anderes Team gewährleistet wird, die Möglichkeit Anwendungen zu erstellen, die konsistent und wartungsfreundlich sind, und die ein Framework verwenden, das von einem Vollzeit-Entwicklungsteam in Kombination mit einer robusten Open-Source-Community unterstützt wird, sind für die meisten Unternehmen Schlüsselprioritäten.

Modularität

Bei Angular geht es darum, Code in „Buckets“ zu organisieren. Alles, was Sie erstellen, ob es sich um Komponenten, Services, Pipes oder Anweisungen handelt, muss in einem oder mehreren Buckets organisiert werden. Wenn Sie in Ihrer Organisation oftmals mit Spaghetti-Codes hantieren müssen, wirkt die Arbeit mit Angular und TypeScript sehr erfrischend. Die „Buckets“, auf die ich mich beziehe, werden in der Angular-Welt „Module“ genannt. Sie bieten eine Möglichkeit, die Anwendungsfunktionalität zu organisieren und in Funktionen und wiederverwendbare Abschnitte zu unterteilen. Module bieten auch viele andere Vorteile, wie z. B. das langsame Laden, wenn eine oder mehrere Anwendungsfunktionen im Hintergrund oder bei Bedarf geladen werden sollen.

Unternehmensanwendungen können sehr groß werden, und die Möglichkeit, die Arbeit auf mehrere Teammitglieder aufzuteilen, und gleichzeitig die Qualität des Codes im Blick zu behalten, ist mit Angular definitiv erreichbar. Module können verwendet werden, um eine Organisation einer Anwendung hinzuzufügen, ähnlich wie Pakete und Namespaces in anderen Sprachen / Frameworks wie Java oder .NET. Ich gebe gerne zu, dass ein solides Verständnis von Angular-Modulen und deren Verwendung wesentlich für einen erfolgreichen Einsatz ist. Doch sobald ein Team Module entsprechend entwickeln kann, schlagen die Vorteile der Arbeitsteilung, der Codekonsistenz, der Produktivität und der Wartung voll durch.

Frühzeitige Fehlererkennung

Angular wurde mit TypeScript erstellt, was viele Vorteile mit sich bringt, wie bspw.::

- TypeScript ist eine Obermenge von JavaScript. TypeScript ist keine eigene eigenständige Sprache wie CoffeeScript, Dart oder andere, und ist daher besonders mächtig. Das heißt, ich kann vorhandenen ES5- oder ES2015+ JavaScript-Code verwenden,

ihn in eine TypeScript-.ts-Datei packen (oder sogar direkt mit der .js-Datei arbeiten), und der Code funktioniert einwandfrei. TypeScript kompiliert / konvertiert den Code einfach bis zu ES5 oder ES2015, je nachdem, was Sie konfigurieren.

- TypeScript unterstützt die wichtigsten ES2015-Funktionen sowie ES2016 / ES2017-Funktionen wie Dekorateure, Async / await und andere. Ich sehe es gerne als ES2015 ++. Unterstützte Funktionen finden Sie unter http://kangax.github.io/compat-table/es6.
- TypeScript bietet Unterstützung für Typen (Grundelemente, Schnittstellen und andere benutzerdefinierte Typen). Ja – es gibt einen Grund, warum TypeScript das Wort „Type“ im Namen hat. Obwohl Typen optional sind, werden sie dringend empfohlen, sofern Sie Fehler im Entwicklungszyklus frühzeitig erkennen möchten, insbesondere bei größeren Anwendungen. Durch sie wird es viel einfacher zu erkennen, wenn etwas übergeben oder falsch verwendet wird.
- TypeScript-Code kann direkt im Browser (oder in einem Editor) debuggt werden, sofern die entsprechenden Zuordnungsdateien während der Erstellungszeit erstellt werden.
- Mit TypeScript können Sie Klassen und/oder funktionale Programmiertechniken verwenden. Sie sind nicht auf eine Möglichkeit beschränkt, und Sie können alle TypeScript-spezifischen Funktionen deaktivieren.

Ich könnte viele andere Funktionen auflisten (unter TypeScript-Site finden Sie weitere), aber letztendlich wird das Angular-Framework mit TypeScript erstellt. Wenn Sie es in Ihren Projekten verwenden (was ich sehr empfehle), können Sie Fehler früh im Entwicklungszyklus oder bei der Durchführung von Wartungsarbeiten identifizieren. In Bezug auf Unternehmensanwendungen bietet TypeScript „Leitplanken“ für Ihren JavaScript-Code. So stellen Sie sicher, dass Entwickler und Teams beim Erstellen von Anwendungen nicht „über die Wupper“ gehen.

Neben den Vorteilen, die TypeScript bietet, wurde Angular mit dem Gedanken an Testbarkeit entwickelt. Die Angular-CLI macht den Prozess des Komponententests und des End-to-End-Tests zum Kinderspiel (es wird standardmäßig Karma und Jasmine für Komponententests verwendet, Sie können jedoch beliebige Test-Tools verwenden). Geben Sie einfach „ng test“ in die Befehlszeile ein und alle Tests im Projekt werden ausgeführt. Der Befehl „ng generate“ generiert automatisch eine Spezifikationsdatei für Sie, wenn Sie eine Komponente, einen Service usw. erstellen. Wenn Ihr Unternehmen Unit-Tests nutzt, ist dies definitiv ein weiterer Vorteil, der Ihnen hilft, Fehler in der Entwicklung frühzeitig zu erkennen.

Zusammenfassung

Es gibt sicherlich viele weitere Vorteile, die diskutiert werden könnten, aber ich wollte mich auf fünf konzentrieren, die ich für wichtig halte. Sind dies die „richtigen“ Vorteile, auf die Sie sich konzentrieren können? Das ist natürlich eine sehr subjektive Frage. Für mein Unternehmen und viele andere Unternehmen, mit denen wir zusammenarbeiten, stehen Konsistenz, Wartbarkeit, Produktivität, Modularität und die Möglichkeit, Fehler frühzeitig zu erkennen, ganz oben auf der Liste.

Es ist wichtig anzumerken, dass viele dieser Vorteile auch mit anderen JavaScript-Bibliotheken / Frameworks erzielt werden können („one size does not fit all“). Das Ziel dieses Beitrags besteht NICHT darin, Sie zu überzeugen, dass Angular das einzige Client-seitige Framework ist, das diese Vorteile bietet. Stattdessen ist es das Ziel, denjenigen, die Angular und TypeScript in Betracht ziehen, dabei zu helfen, einige der wichtigsten Vorteile zu verstehen.

Hinweis:

Der Beitrag von Dan Wahlin ist im Original unter https://blog.codewithdan.com/5-key-benefits-of-angular-and-typescript/ erschienen. Mit Zustimmung von Dan Wahlin übersetzen wir verschiedene Beiträge von ihm hier in unserem Blog vom Englischen ins Deutsche. Beiträge im Original finden Sie unter https://blog.codewithdan.com/. In LinkedIn können Sie sich mit ihm unter https://www.linkedin.com/in/danwahlin/ vernetzen.

Dan Wahlin

Dan Wahlin ist Entwickler, Architekt, Technologietrainer, Autor und öffentlicher Redner mit Fachkenntnissen in der Architektur, dem Design und dem Bau von lokal und in der Cloud gehosteten Webanwendungen unter Verwendung verschiedener Technologien (JavaScript, TypeScript, Angular, .NET, C #, ASP.NET) , Node.js, HTML5 / CSS, Docker, Azure, Github / git, Windows / OSX / Linux und mehr). Er ist Autor mehrerer Bücher, verfasste Hunderte von technischen Artikeln und erstellt Videokurse für Pluralsight und Udemy. Dan spricht gerne

bei Anwendergruppen, Meetups, Code-Camps und Konferenzen auf der ganzen Welt, darunter Microsoft BUILD und (früher) TechEd, MIX, DevIntersection, AngleBrackets, DevSum, Ng-Conf, Angular U und viele andere.

Kommentar absenden

Deine E-Mail-Adresse wird nicht veröffentlicht. Erforderliche Felder sind mit * markiert.

Kommentar

Name *

E-Mail *

Website

Kommentar absenden