MOBILE APPS MIT NATIVESCRIPT UND ANGULAR 2 ENTWICKELN

# Das Beste aus allen Welten

Wie man native Apps für Android und iOS mit NativeScript entwickelt.

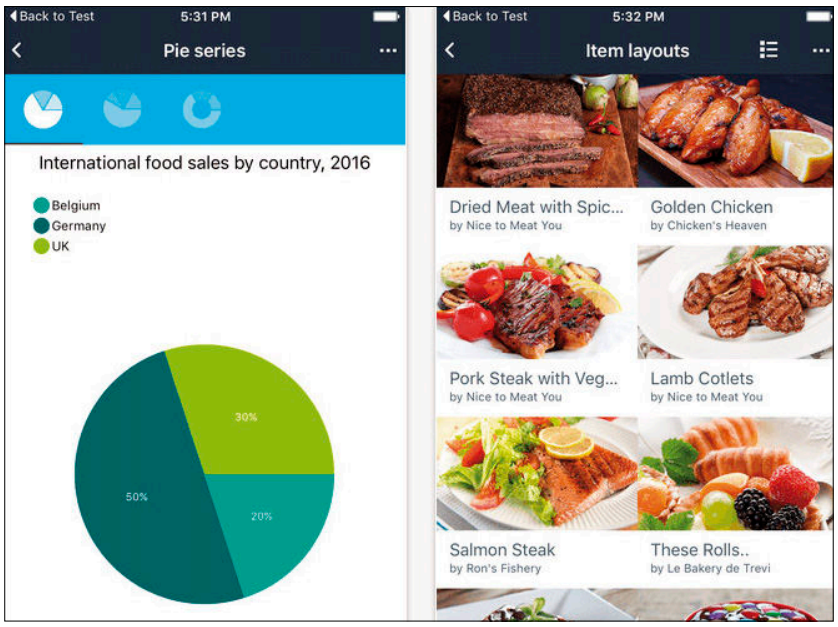
Die Anforderungen an moderne Apps sind unter anderem eine ansprechende Ästhetik, ein plattformspezifisches Nutzererlebnis und bestmögliche Performance. Normalerweise werden hierzu eigenständige Apps für die mobilen Betriebssysteme erstellt. Doch parallele Entwicklungen erzeugen gleichzeitig erhöhte Kosten.

Eine Antwort darauf sind hybride Apps auf Basis von HTML und JavaScript. Das bedeutet automatisch aber ein paar technische Beschränkungen. Durch NativeScript von Progress, ist es möglich, native Apps direkt mit JavaScript zu entwickeln.

Diese Apps lassen sich nicht mehr von Lösungen unterscheiden, die klassisch auf Basis von Objective-C beziehungsweise Swift oder Java entwickelt worden sind (Bild 1). Sie verfügen über eine beachtliche Geschwindigkeit und verwenden die normalen Bedienelemente des jeweiligen mobilen Betriebssystems.

## Was ist NativeScript?

NativeScript ist ein Open-Source-Framework zur Entwicklung mobiler Apps [1]. Neben JavaScript wird auch die JavaScript-Obermenge TypeScript direkt unterstützt. Aktuell stehen als Zielplattformen sowohl Android als auch iOS zur Verfügung. Seit Version 1.7 (März 2016) ist auch ein Support für UWP-Apps (Universal Windows Platform) hinzugekommen. UWP-Apps sind auf Windows 10 Mobile und Windows 10 ausführbar. Momentan wird die neueste Zielplattform aber noch als „proof of concept“ eingestuft. Derzeit sind somit nur NativeScript-Apps für Android und iOS reif für den produktiven Einsatz.




**Zwei NativeScript-Screens** unter iOS (Bild 1)

Auf den ersten flüchtigen Blick scheint das Framework eine weitere Variante des hybriden Ansatzes zu sein. Das bekannteste hybride Framework ist wohl Apache Cordova [2], das den Einsatz von HTML und JavaScript zur App-Entwicklung salonfähig gemacht hat. Cordova lässt Webanwendungen in einem Browser laufen und ermöglicht über Schnittstellen den Zugriff auf native Funktionen.

Sofern Sie mit dem SPA-Framework AngularJS vertraut sind, steht Ihnen zum Beispiel durch Einsatz des Ionic-Frameworks eine populäre und weit verbreitete Lösung zur Verfügung. Durch den Einsatz von vorbereiteten Themes und eigenem grafischen Geschick muten die fertigen Apps anschließend wie native Anwendungen an. Ein grundlegendes Problem ergibt sich jedoch prinzipiell immer: Eine hybride App ist und