



Introducing NativeScript

TJ VanToll | @tjvantoll

nativescript.org



NativeScript

Build truly native apps with JavaScript

Develop iOS, Android and Windows Phone apps from a single code base

GET STARTED



VIEW IN GITHUB



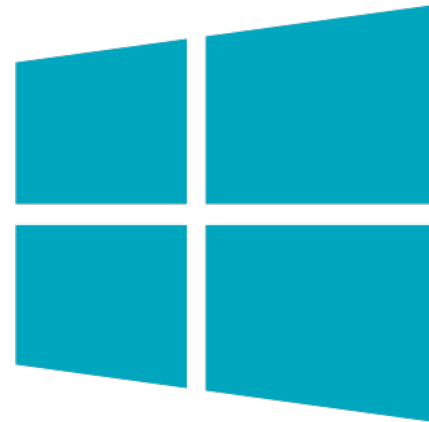
NativeScript Timeline

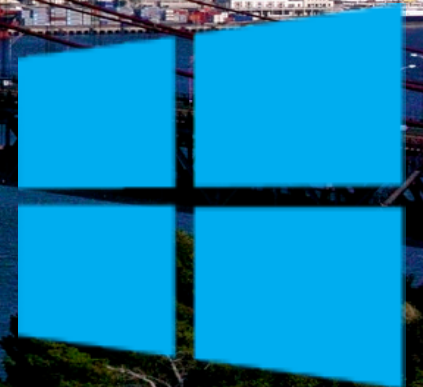
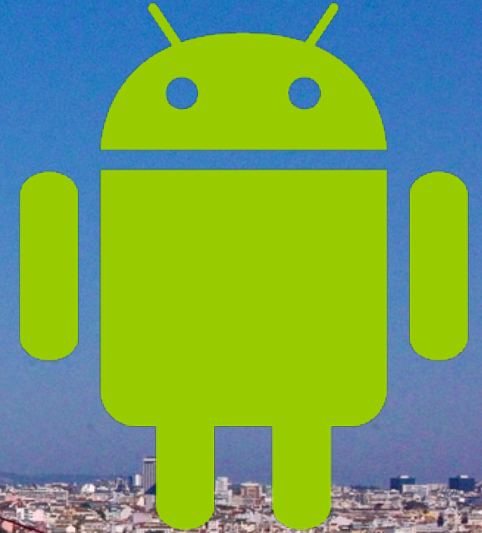
- 0.9
 - Public Beta
 - March 5th, 2015
- 1.0
 - Go-live license
 - Windows Phone support
 - May 2015



What is NativeScript?

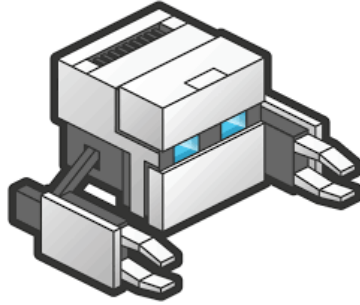
- A runtime for building and running *native* iOS, Android, and Windows Phone apps with a single, JavaScript code base







!=



- No DOM



!=



- No cross compilation



NativeScript Android example

```
var time = new android.text.format.Time();  
time.set( 1, 0, 2015 );  
console.log( time.format( "%D" ) );
```

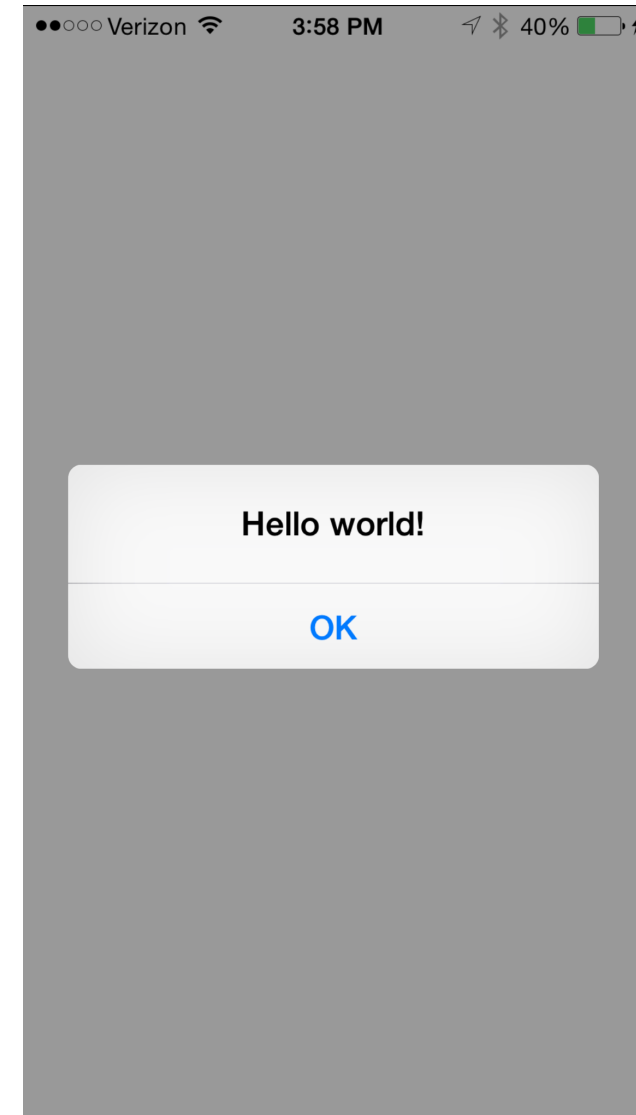
Output: "01/01/15"





NativeScript iOS example

```
var alert = new UIAlertView();  
alert.message = "Hello world!";  
alert.addWithTitle( "OK" );  
alert.show();
```



UIAlertView Class Reference

Apple Inc. [US] https://developer.apple.com/library/prerelease/ios/documentation/UIKit/Reference/UIAlertView_C...

IOS Developer Library — Pre-Release

UIKit Framework Reference > UIAlertView Class Reference

Search iOS Developer Library

Language: [Swift](#) [Objective-C](#) [Both](#) On This Page Options

UIAlertView

Setting Properties

[delegate](#) *Property*

[alertViewStyle](#) *Property*

[title](#) *Property*

[message](#) *Property*

[visible](#) *Property*

Configuring Buttons

[- addButtonWithTitle:](#)

[numberOfButtons](#) *Property*

[- buttonTitleAtIndex:](#)

[- textFieldAtIndex:](#)

[cancelButtonIndex](#) *Property*

[firstOtherButtonIndex](#) *Property*

Displaying

[- show](#)

Tasks

- Creating Alert Views
- Setting Properties
- Configuring Buttons
- Displaying
- Dismissing

Constants

- UIAlertViewStyle

Related Documentation

- Alert Views in UIKit User Interface Catalog

Related Sample Code

- AdvancedURLConnections
- GKTapper
- MVCNetworking
- SquareCam
- URLCache

Feedback

```
var alert = new UIAlertView();
alert.message = "Hello world!";
alert.addButtonWithTitle( "OK" );
alert.show();
```



How does this work?



NativeScript and JS VMs

- NativeScript runs JavaScript on a JavaScript VM
 - JavaScriptCore on iOS
 - V8 on Android
 - JavaScriptCore on Windows



```
var time = new android.text.format.Time();  
time.set( 1, 0, 2015 );  
console.log( time.format( "%D" ) );
```

- Runs on V8

```
var alert = new UIAlertView();  
alert.message = "Hello world!";  
alert.addButtonWithTitle( "OK" );  
alert.show();
```

- Runs on JavaScriptCore



Gathering Native APIs

- NativeScript uses reflection to build a list of available APIs for each platform.
- For optimal performance, this metadata is pre-generated, and injected into the app package at build time.



Injecting native APIs

- V8/JavaScript Core have APIs to inject global variables

v8 Namespace Reference

Debugger support for the V8 JavaScript engine. [More...](#)

Namespaces

namespace	internal
-----------	----------

Data Structures

class	AccessorInfo The information passed to an accessor callback about the context of the property access. More...
class	ActivityControl An interface for reporting progress and controlling long-running activities. More...
class	Arguments The argument information given to function call callbacks. More...
class	Array An instance of the built-in array constructor (ECMA-262, 15.4.2). More...
class	Boolean A primitive boolean value (ECMA-262, 4.3.14). More...
class	BooleanObject A Boolean object (ECMA-262, 4.3.15). More...
class	Context A sandboxed execution context with its own set of built-in objects and functions. More...
class	CpuProfile CpuProfile contains a CPU profile in a form of two call trees: <ul style="list-style-type: none">• top-down (from main() down to functions that do all the work);• bottom-up call graph (in backward direction).



Invoking native APIs

```
var time = new android.text.format.Time();
```

- V8/JavaScriptCore have C++ callbacks for JS function calls and property accesses.
- The NativeScript runtime uses those callbacks to translate JS calls into native calls.
- On iOS, you can directly call Objective-C APIs from C++ code.
- On Android, NativeScript uses Android's JNI (Java Native Interface) to make the bridge from C++ to Java.



```
var time = new android.text.format.Time();
```

- 1) The V8 function callback runs.
- 2) The NativeScript runtime uses its metadata to know that Time() means it needs to instantiate an android.text.format.Time object.
- 3) The NativeScript runtime uses the JNI to instantiate an android.text.format.Time object and keeps a reference to it.



- 4) The NativeScript runtime returns a JS object that proxies the Java Time object.
- 5) Control returns to JS where the proxy object gets stored as a local time variable.

```
var time = new android.text.format.Time();  
time.set( 1, 0, 2015 );  
console.log( time.format( "%D" ) );
```





**So do you only write native
code?**

No



TNS modules

- NativeScript-provided modules that provide cross-platform functionality.
- There are dozens of them and they're easy to write yourself.
- TNS modules follow Node module's conventions (CommonJS).



TNS file module

```
var fileSystemModule = require( "file-system" );  
new fileSystemModule.File( path );
```



`new java.io.File(path);`



`NSFileManager defaultManager();`
`fileManager.createFileAtPathContentsAttributes(path);`

HTTP module example

```
var http = require( "http" );  
http.getJSON( "https://api.myservice.com" )  
  .then(function( result ) {  
    // result is JSON Object  
  });
```



Custom TNS modules

```
// device.ios.js
module.exports = {
    version: UIDevice.currentDevice().systemVersion
}
```

```
// device.android.js
module.exports = {
    version: android.os.Build.VERSION.RELEASE
}
```



Using the custom device module

```
var device = require( "./device" );  
console.log( device.version );
```



Community modules

- <https://github.com/alejonext/NativeNumber>
 - Someone created this 7 hours after the NativeScript public release.



But how do I turn this into an app?



Two ways to use NativeScript

1)  **Telerik**PlatformSM

2) `npm install -g nativescript`





Telerik PlatformSM

<http://telerik.com/platform>

- Backend-as-a-service
 - Push notifications, cloud data, file storage, and more
- Analytics
- AppBuilder
 - Cloud builds (build iOS apps on Windows, Windows Phone apps on a Mac)
 - NativeScript debugging and tooling
- Automated app testing
- And more!





<https://www.telerik.com/purchase/platform>

Telerik Platform 30 Day Trial

FREE

Start now

Try everything Telerik Platform
has to offer, FREE, for 30 days

 **All Platform Services**

 **Web, Hybrid & Native UI**

Unlimited trial support


Telerik Platform Developer

\$39 /month/user
requires annual upfront payment

Subscribe

Ideal for tinkerers and hobbyists just
getting started with mobile app
development

 **Core Platform**

 **Hybrid UI**

Limited web support

Telerik Platform Professional

\$79 /month/user
requires annual upfront payment

Subscribe

For professional developers and small
teams building full-featured employee and
consumer apps

 **Core Platform**

+ Advanced Cloud Services
+ Direct App Store Deployment

 **Hybrid & Native UI**

Limited web support

**MOST
POPULAR**

Telerik Platform Business


\$149 /month/user
requires annual upfront payment

Subscribe

For developers and large teams
building advanced apps
connected to business data

 **Pro Platform**

+ Active Directory Integration
+ Enterprise Data Connectors
+ Private App Distribution

 **Web, Hybrid & Native UI**

Unlimited web support

NativeScript CLI

- Free and open source
- <https://github.com/nativescript/nativescript-cli>



NativeScript CLI requirements

- <https://github.com/nativescript/nativescript-cli#system-requirements>



- JDK, Apache Ant, Android SDK



- Xcode, Xcode CLI tools, iOS SDK



Starting a new project

```
$ npm install -g nativescript
```

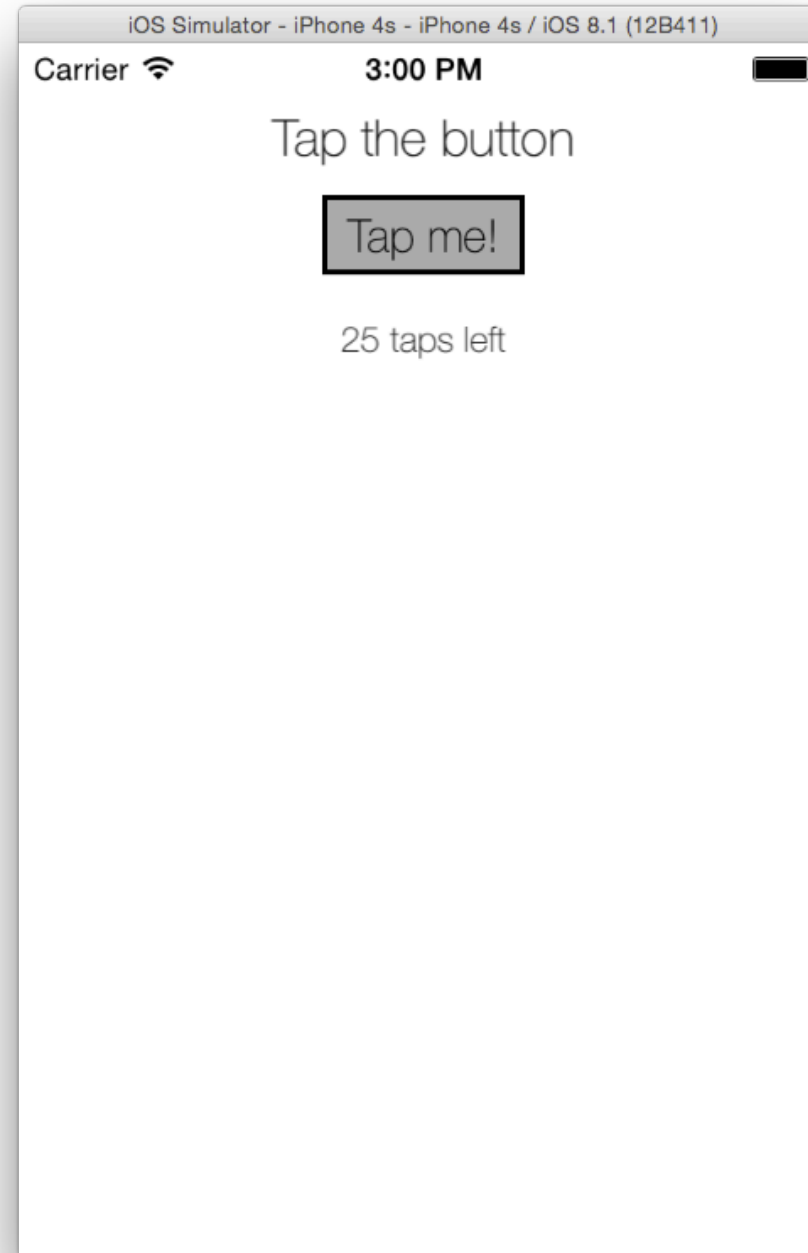
```
$ tns create hello-world
```

```
$ cd hello-world
```



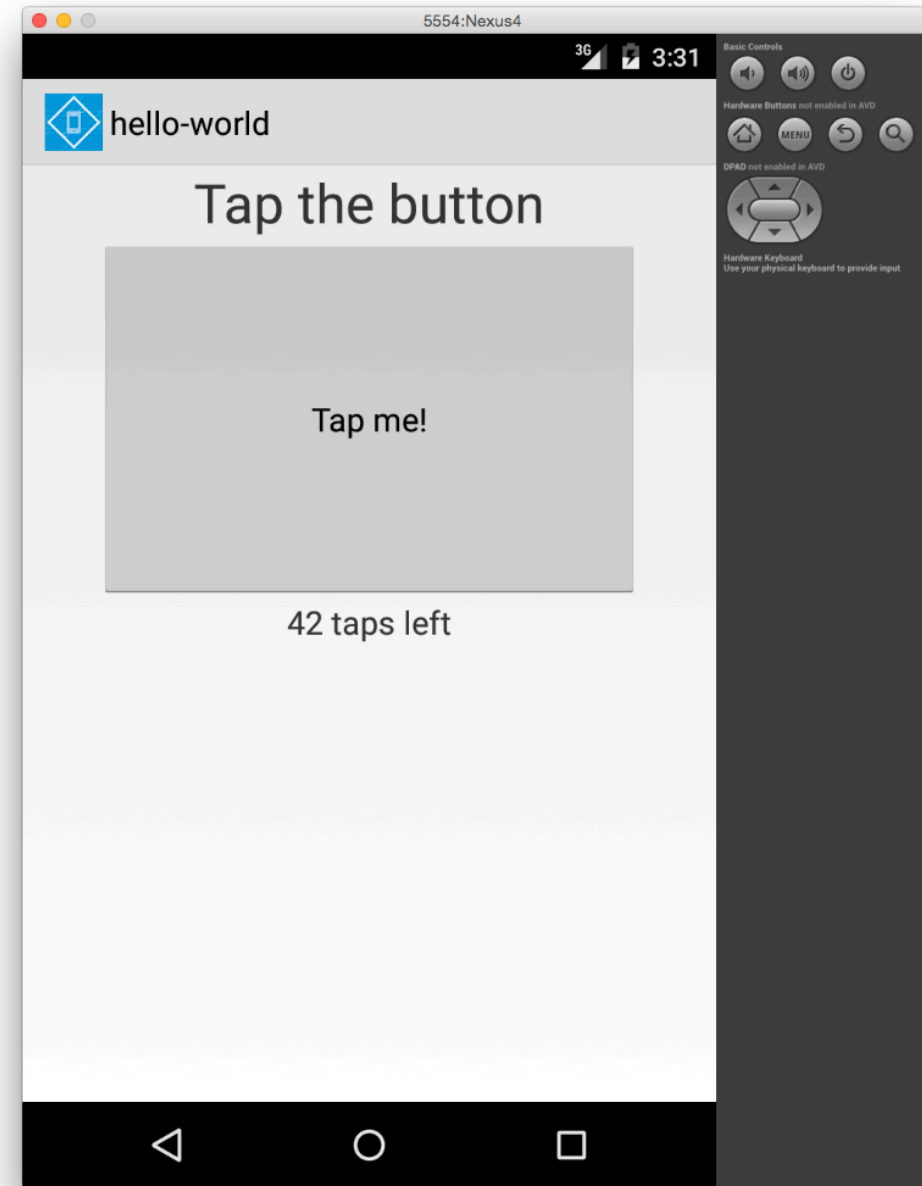
Running on iOS

```
$ tns platform add ios  
$ tns run ios --emulator
```



Running on Android

```
$ tns platform add android  
$ tns run android --emulator
```



```
.
└─ name-of-app
    └─ app
        └─ app
            └─ app.css      <-- app styling
            └─ app.js       <-- app starting point
            └─ main-page.css
            └─ main-page.js
            └─ main-page.xml
            └─ node_modules <-- npm modules
                └─ ...
        └─ App_Resources    <-- icons, splash screens, config files
            └─ ...
        └─ tns_modules      <-- NativeScript modules
            └─ ...
    └─ platforms
        └─ android
        └─ ios
```



app.js

```
var application = require( "application" );  
application.mainModule = "main-page";  
application.start();
```



Pages

- XML markup structure
- Elements (e.g. `<Page>`, `<Label>`) are TNS modules

```
<Page>  
    <Label text="hello world" />  
</Page>
```



Data binding

```
<Page loaded="load">  
  <Label text="hello world" />  
</Page>
```

```
exports.load = function( args ) {  
  args.object.bindingContext = { message: "hello world" };  
}
```



Data binding improved

```
var observableModule = require( "data/observable" );

exports.load = function( args ) {
    var data = new observableModule.Observable();
    data.set( "message" , "hello world" );
    args.object.bindingContext = data;
}
```



CSS

```
Label {  
    color: red;  
    font-size: 20;  
    width: 200;  
    margin: 20;  
}
```



<http://docs.nativescript.org/styling#supported-properties>

Supported Properties

This is the list of the properties that can be set in CSS or through the style property of each View:

CSS Property	JavaScript Property	Description
color	color	Sets a solid-color value to the matched view's foreground.
background-color	backgroundColor	Sets a solid-color value to the matched view's background.
font-size	fontSize	Sets the font size of the matched view (only supports



Demo time!



Contribute!

(nativescript.org/contribute)

Contributing to NativeScript

Thank you for your interest in contributing to the NativeScript project!

Anyone wishing to contribute to the NativeScript project MUST read & sign the NativeScript [Contribution License Agreement](#). The NativeScript team cannot accept pull requests from users who have not signed the CLA first.

NativeScript is a complex framework, involving cross-platform modules, a Command-Line Interface and platform-specific runtimes. Each of these follows a specific technology, therefore the contribution instructions are different for each.

Please, visit these repositories for detailed contribution guidelines:

[Cross-Platform modules](#)

[Command-Line Interface](#)

[Android-Runtime](#)

[iOS-Runtime](#)



Follow NativeScript



- @nativescript
- <https://nativescript.org/blog>



Questions?

- TJ VanToll | @tjvantoll



Thanks!

