



Introducing NativeScript

[Your name here]

What is NativeScript?

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



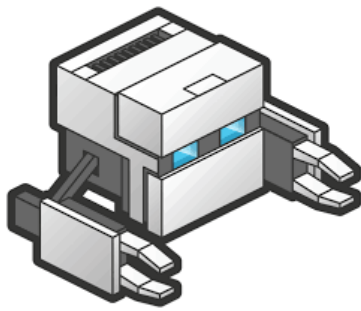
Not sure...

if this is PhoneGap





!=



- No DOM



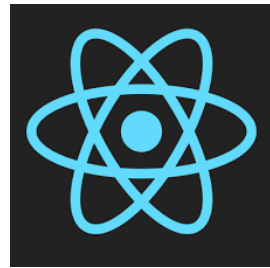
!=



- No cross compilation



!=



- Direct access to native APIs in JS



Why NativeScript?

- Skills reuse
 - Standards-based JavaScript, CSS, optionally TypeScript
- Code reuse
 - npm modules, 3rd-party iOS and Android libraries
- Easily use native APIs
 - No wrappers to access native APIs
 - Use native UI elements
- Open source!



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:

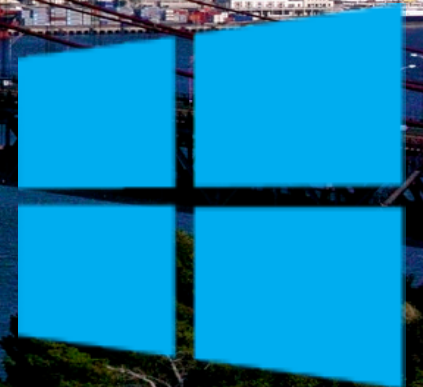
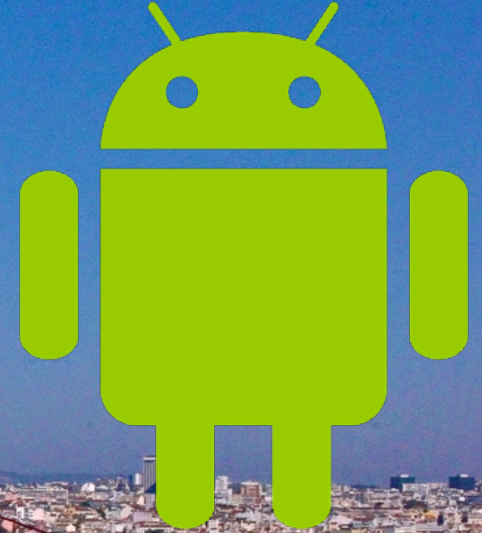
[Cross-Platform modules](#)

[Command-Line Interface](#)

[Android-Runtime](#)

[iOS-Runtime](#)





NativeScript Android example

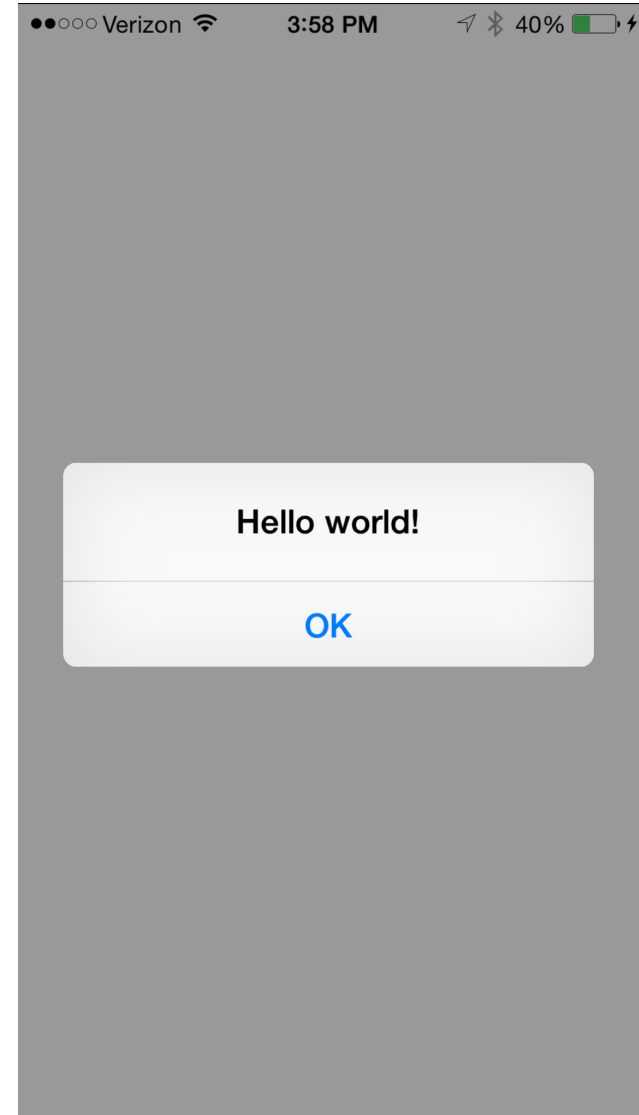
```
var time = new android.text.format.Time();  
time.set( 1, 0, 2015 );  
console.log( time.format( "%D" ) );
```

Output: "01/01/15"



NativeScript iOS example

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



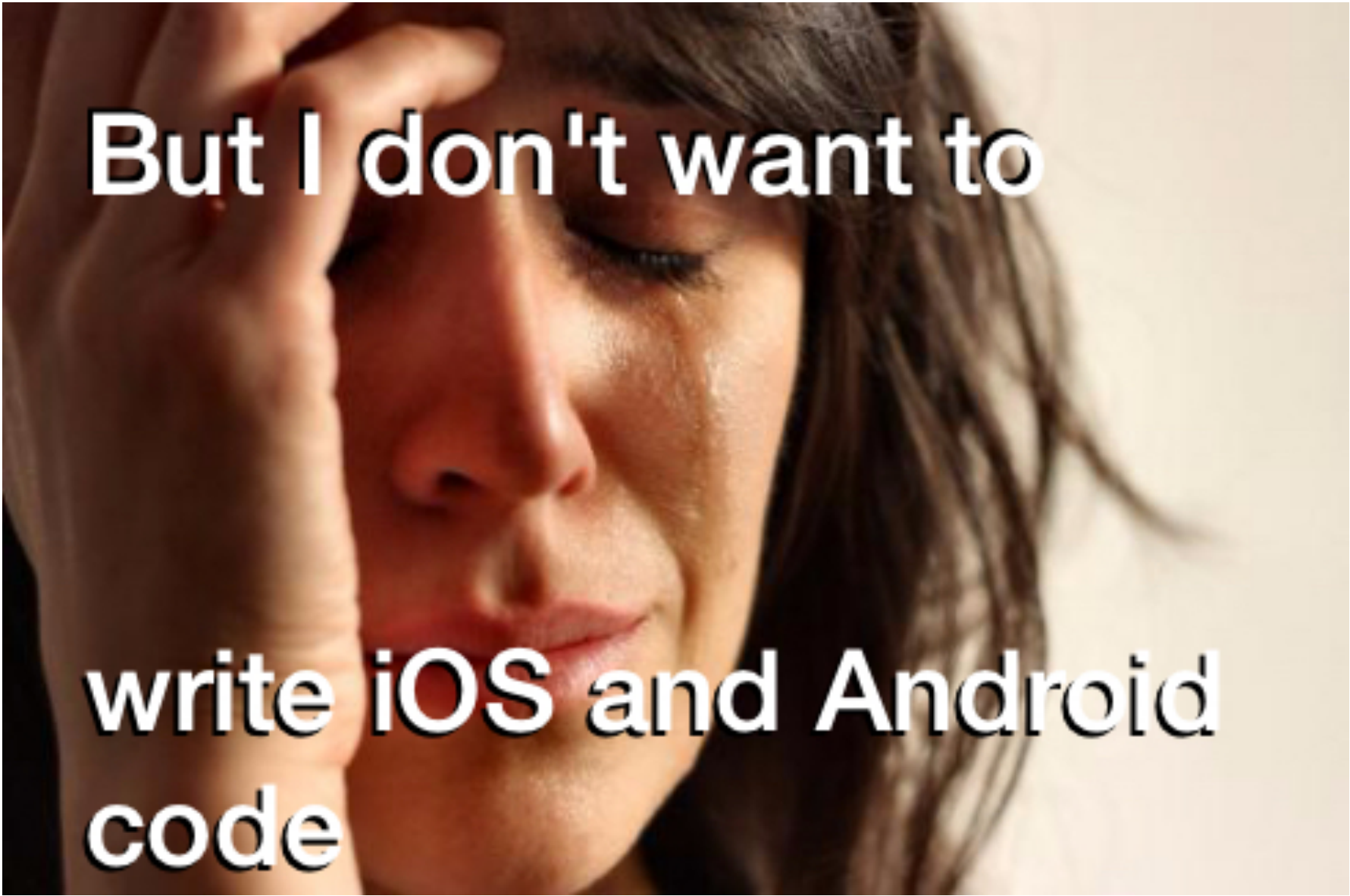

```
var time = new android.text.format.Time();  
time.set( 1, 0, 2015 );  
console.log( time.format( "%D" ) );
```

- Runs on V8

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

- Runs on JavaScriptCore



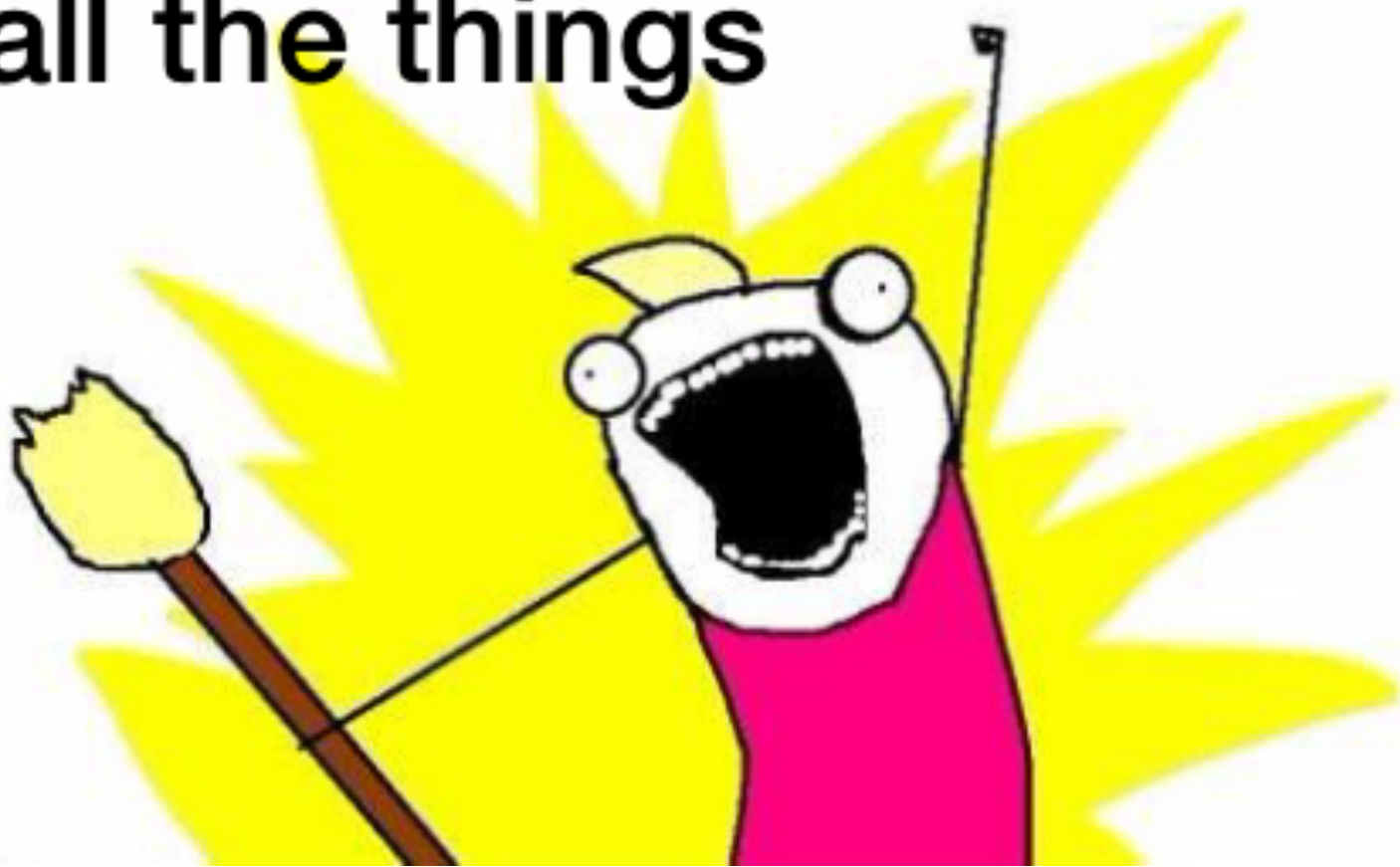


But I don't want to

**write iOS and Android
code**



NativeScript modules for all the things



NativeScript modules

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



NativeScript 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  
  });
```



Community modules



nativescript

Command-line interface for building NativeScript pr...

version 0.10.0

305 downloads in the last week



nativescript-sqlite

A sqlite NativeScript module for Android and (soon) iOS

version 0.0.2

21 downloads in the last week



nativescript-maps

A NativeScript module for using native map APIs

version 0.1.1

39 downloads in the last week



nativescript-texttospeech

A texttospeech NativeScript module for Android and ...

version 1.0.1

0 downloads in the last week



nativescript-flashlight

A flashlight NativeScript module for Android and iOS

version 0.1.1

180 downloads in the last week



nativescript-vibrate

A vibrate NativeScript module for Android and iOS

version 1.0.1

22 downloads in the last week



nativescript-phone

A phone NativeScript module for Android and iOS

version 0.1.2

5 downloads in the last week

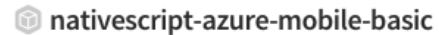


nativescript-social-share

A NativeScript module to use the native social sharin...

version 0.1.0

0 downloads in the last week



nativescript-azure-mobile-basic

A NativeScript module to read Azure Mobile Services ...

version 0.1.2

15 downloads in the last week



tns-ios

Telerik NativeScript Runtime for iOS

version 0.10.0

156 downloads in the last week



tns-template-hello-world

Hello World project template for NativeScript

version 0.10.1

279 downloads in the last week



nativenumber

Is a creator to native numbers in Java

version 0.1.2

4 downloads in the last week



tns-android

NativeScript Runtime for Android

version 0.10.0

189 downloads in the last week



ios-sim-portable

ios-sim-portable =====

version 1.0.6

450 downloads in the last week



appbuilder

command line interface to Telerik AppBuilder

version 2.8.3-331

232 downloads in the last week

<https://www.npmjs.com/search?q=nativescript>



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, etc
- Analytics
- AppBuilder
 - Cloud builds (build iOS apps on Windows, Windows Phone apps on a Mac)
 - NativeScript debugging and tooling
- Screen Builder (app scaffolding)
- And more!



Telerik AppBuilder IDE Options

- In-Browser Client
- Visual Studio Extension
- Sublime Text Package
- Command-Line Interface





<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

Or, use AppBuilder individually

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

<p>Telerik Platform 30 Day Trial</p> <p>FREE</p> <p>Unlimited projects</p> <p>Start for Free</p>	<p>MOST POPULAR</p> <p>AppBuilder Developer</p> <p>\$19 /month/user requires annual upfront payment</p> <p>5 projects</p> <p>Subscribe</p>	<p>AppBuilder Professional</p> <p>\$49 /month/user requires annual upfront payment</p> <p>Unlimited projects</p> <p>Subscribe</p>	<p>AppBuilder Business</p> <p>\$99 /month/user requires annual upfront payment</p> <p>Unlimited projects</p> <p>Subscribe</p>
---	--	--	--



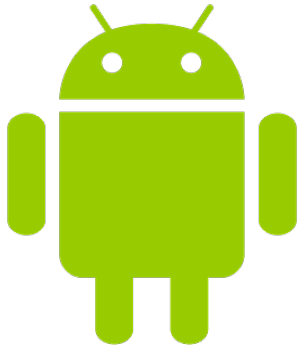
NativeScript CLI

- 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
```

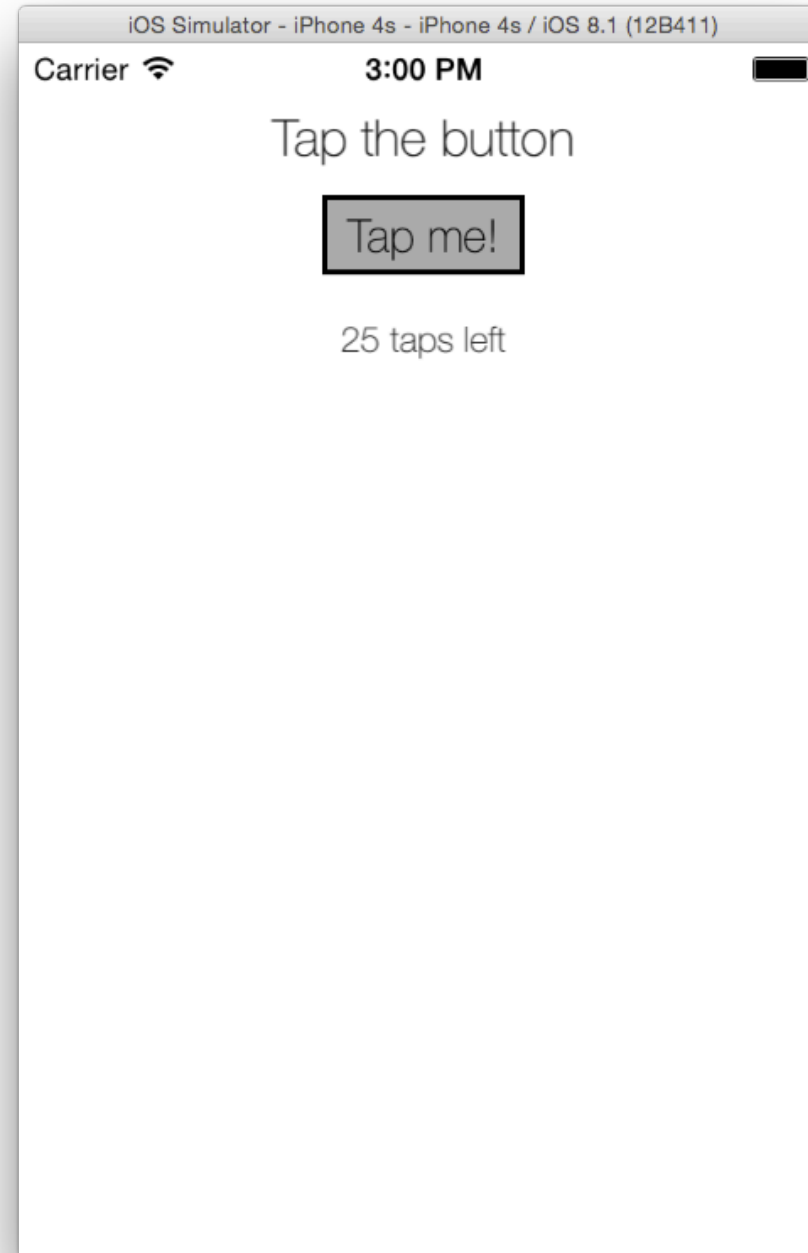
```
$ tns create hello-world
```

```
$ cd hello-world
```



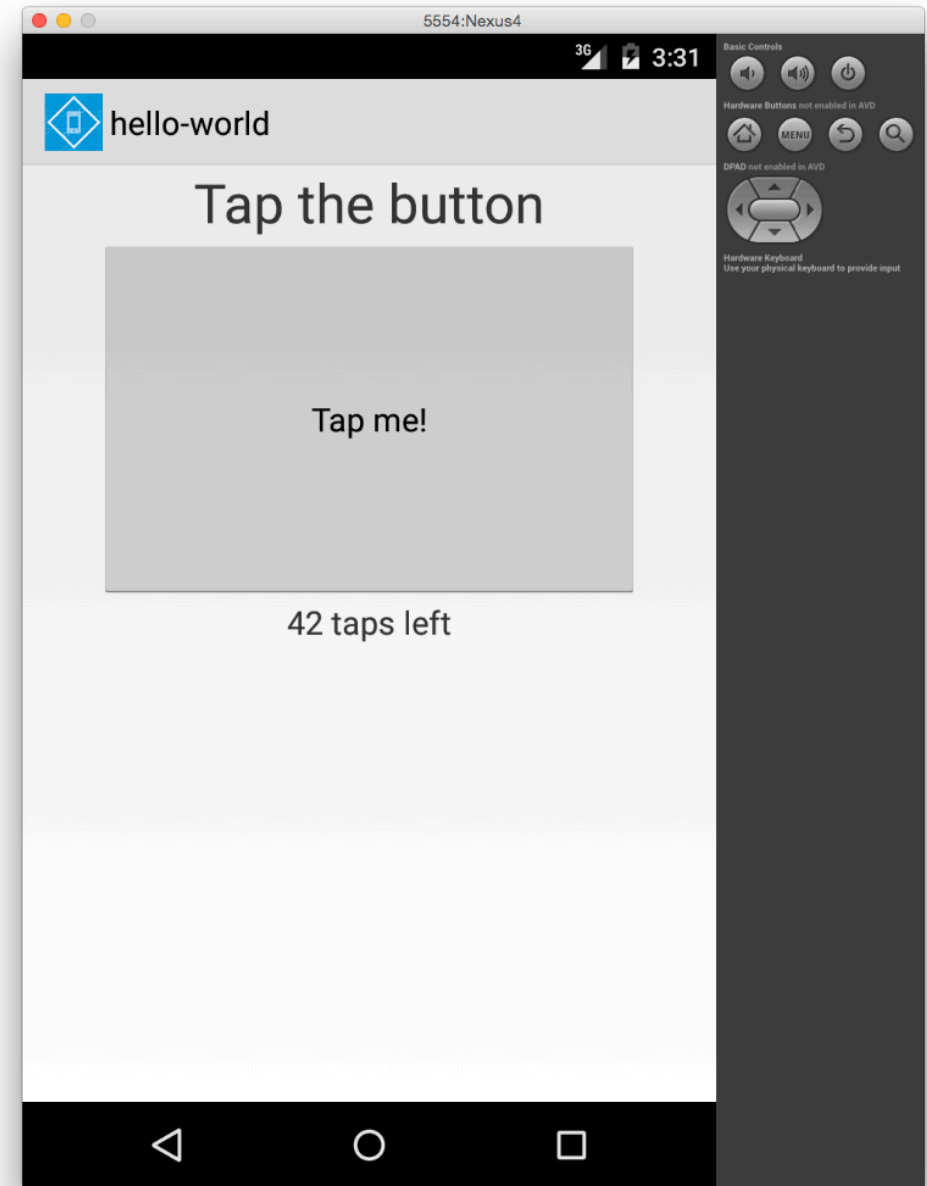
Running on iOS

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



Running on Android

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



```
.
├── app
│   ├── App_Resources <-- icons, splash screens, config files
│   │   └── ...
│   ├── app.css <-- App styling
│   ├── app.js <-- App starting point
│   ├── main-page.css
│   ├── main-page.js
│   ├── main-page.xml
│   ├── node_modules <-- npm modules
│   │   └── ...
│   ├── package.json
│   └── 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 NativeScript modules

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



Custom XML Components

Example: Code-Only Custom Component

This sample `main-page.xml` uses two custom components defined in separate declarations in the `xml-declaration` directory. The custom controls are wrapped horizontally.

XML

```
<Page
  xmlns:customControls="app/xml-declaration/mymodule"
  xmlns:customOtherControls="app/xml-declaration/mymodulewithxml">
  <WrapLayout>
    <customControls:MyControl />
    <customOtherControls:MyControl />
  </WrapLayout>
</Page>
```

<http://docs.nativescript.org/ui-with-xml#custom-components>



Data binding

```
<Page loaded="load">  
  <Label text="{{ message }}" />  
</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



Hands-on Lab

- <https://github.com/tjvantoll/summer-of-nativescript/blob/master/july/lab.md>



Follow NativeScript



- nativescript.org
- [@nativescript](https://twitter.com/nativescript)
- nativescript.org/blog

