

By



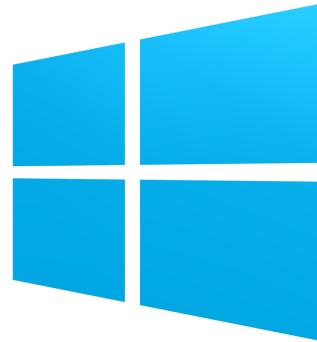
Let's talk NativeScript

This changes everything!

So you want to build a Mobile App?



Your zen is lost quickly ..



Too many platforms

Too many pitfalls with **Native** or **Hybrid** Apps

Confused Developer = Sad Panda!



So why NativeScript? {N}



Truly Native Apps with
JavaScript

Wait .. What?



- Native iOS / Andoid / Windows Phone* Apps
- Built with JavaScript + CSS + XML
- Single Code Base

You're Welcome!

*Coming Soon



Tell me more ..

100% Access to Native Platform API

The entire native platform functionality is available in the JavaScript layer

The screenshot shows a code editor interface with three tabs open:

- button.ios.js**: Contains code for creating a Windows Button control.
- main.js**: Contains a single line of code creating a Windows Button.
- button.android.js**: Contains code for creating an Android Button control.

```
button.ios.js
var Button = (function (_super) {
    __extends(Button, _super);
    function Button() {
        var _this = this;
        _super.call(this);
        this._ios = UIButton.buttonWithType(UIButtonType.U);
        this._clickHandler = ClickHandlerClass.alloc();
        this._clickHandler[OWNER] = new WeakRef(this);
        this._ios.addTargetActionForControlEvents(this._clickHandler, UIControlEvent.TouchUpInside, this._stateChangedHandler);
        this._stateChangedHandler = new stateChanged.ControlStateHandler();
        _this._goToVisualState(s);
    });
    Object.defineProperty(Button.prototype, "ios", {
        get: function () {
            return this._ios;
        },
        enumerable: true,
        configurable: true
    });
    return Button;
})(common.Button);

main.js
var wpButton = new System.Windows.Controls.Button();
btn.Width = 148;
btn.Height = 148;
wpButton.Content = "click me";

button.android.js
Button.prototype._createUI = function () {
    var that = new WeakRef(this);
    var buttonExtends = android.widget.Button.extend({
        drawableStateChanged: function () {
            this.super.drawableStateChanged();
        }
    });
    this._android = new buttonExtends(this._context);
    this._android.setOnClickListener(new android.view.View.OnClickListener() {
        onClick: function (v) {
            var owner = that.get();
            if (owner) {
                owner._emit(common.knownEvents.click);
            }
        }
    });
}
```

Standard-based ECMAScript5+ JavaScript and CSS

No need for rocket science, use your existing skills to build native apps



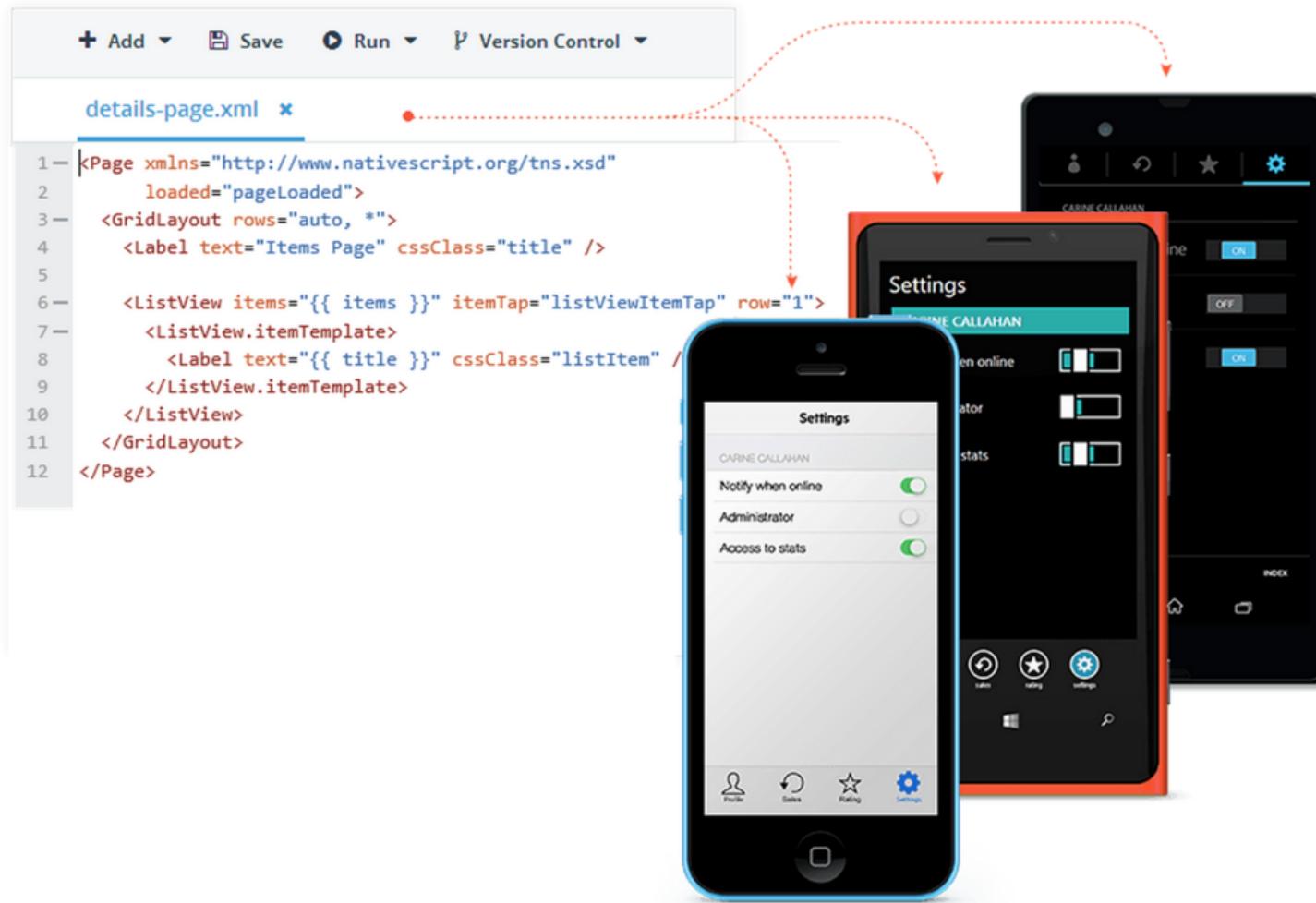
```
+ Add Save Run Version Control
style.css *
1 .title {
2     font-size: 20;
3 }
4
5 .author {
6     font-size: 14;
7 }
8
9 .comments {
10    color: #10C2B0;
11    font-size: 14;
12 }
13
14 .gridpanel {
15     background-color: #363940;
16 }
17

+ Add Save Run Version Control
app.js *
1 (function () {
2
3     // store a reference to the application object that we
4     // later on so that we can use it if need be
5     var app;
6
7     // create an object to store the models for each view
8     window.app = {
9         models: {
10             expenses: {
11                 title: 'Expenses by Category'
12             },
13             settings: {
14                 title: 'Settings'
15             },
16             add: {←→}
17         }
18     };
19 });


```

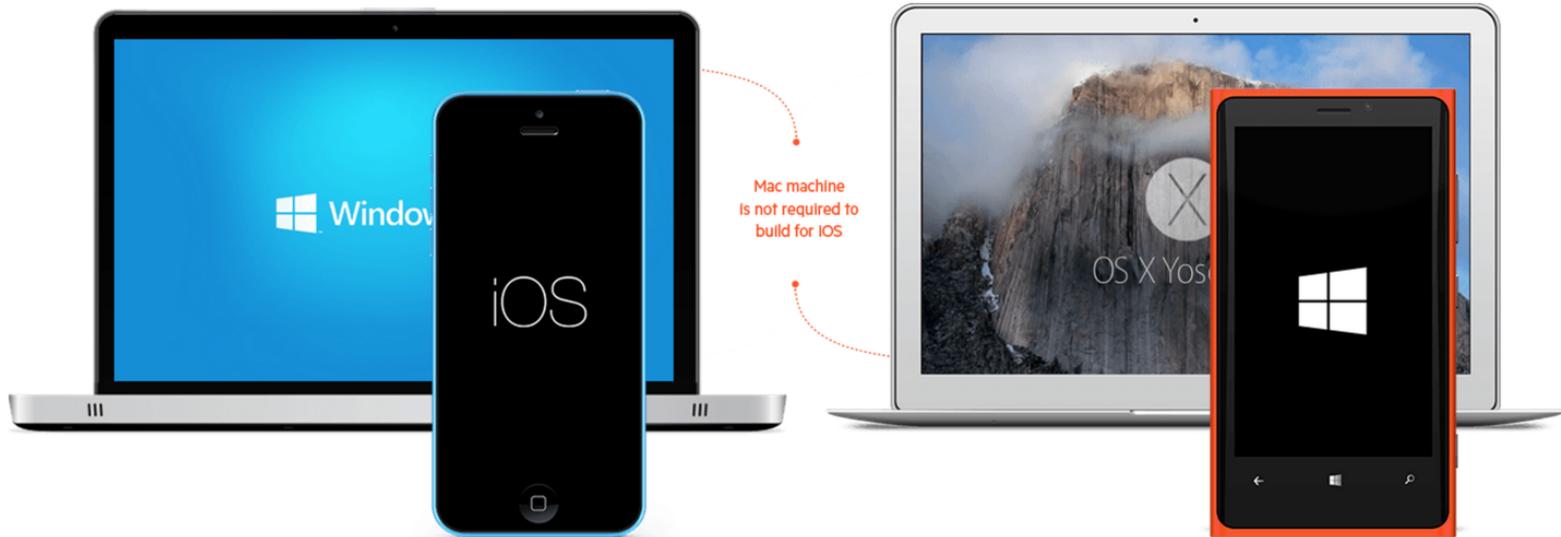
100% Code Sharing

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



Use The Hardware You Already Have

With the existing tools you already have





This is mind blowing ..

Yeah, it's careful orchestration.

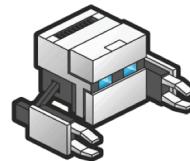
Let's unpack the **Magic!**

How is this working?

{N} is built from ground up



No DOM



No Cross-Compilation



No Plugins Required



PhoneGap

{N} is straight-up JS | Running as Native App

Under the hood

{N} works over an abstraction - a very smart one



Runs JavaScript in a Virtual Machine



JavaScriptCore VM on iOS



V8 VM on Android



JavaScriptCore* VM on Windows Phone

*Tentative

You write JavaScript

{N} utilizes a bridge



Has full access to Native APIs

- That's **all** of iOS + Android APIs!



Uses Reflection to look up Native APIs



List of APIs for each Platform



Metadata pre-generated



Injected into App package @ Build time

Plug & Play?

{N} is very flexible



Allows reuse of skills & assets



Use native libraries for each platform



Use JS libraries without DOM dependency



Shared UI styles through CSS



Full TypeScript Support



Think I'm sold .. where to start?

{N} gives you flexibility

You get **2 ways** to start and **choice of IDEs!**

We LOVE Open Source

Choice #1: {N} is OSS on GitHub



Start @ <https://www.nativescript.org/>



Check out | Use it | Fork it

<https://github.com/NativeScript/NativeScript>

Yep, it's completely **FREE** .. there is no catch!

How do I start?

{N} Command Line Interface makes it easy

Grab the NativeScript CLI:

```
npm install -g nativescript
```

Create Project & Add Platforms:

```
tns create MyApp  
tns platform add android
```

Run Project on Device or Emulator:

```
tns run android  
tns run android --emulator
```

Choice of IDE?

{N} aims to give you flexibility



Sublime Text | With complete Workflow



VS Code | Best for TypeScript



Most other JS/CSS text editors

Use Telerik Platform

Choice #2: {N} is best with Telerik AppBuilder



- Start @ <http://platform.telerik.com>!

Complete **end-to-end** Mobile developer Platform!

What's in Telerik Platform?

Everything you need to build a Mobile app

1. AppPrototyper
2. AppBuilder
3. Backend Services
4. Mobile Testing
5. AppManager
6. AppFeedback
7. Analytics
8. Modulus



You'll love AppBuilder with {N}

All the pieces nicely wrapped up in one **Platform!**

{N} in AppBuilder

Think of it as a friend with benefits

+ Create project

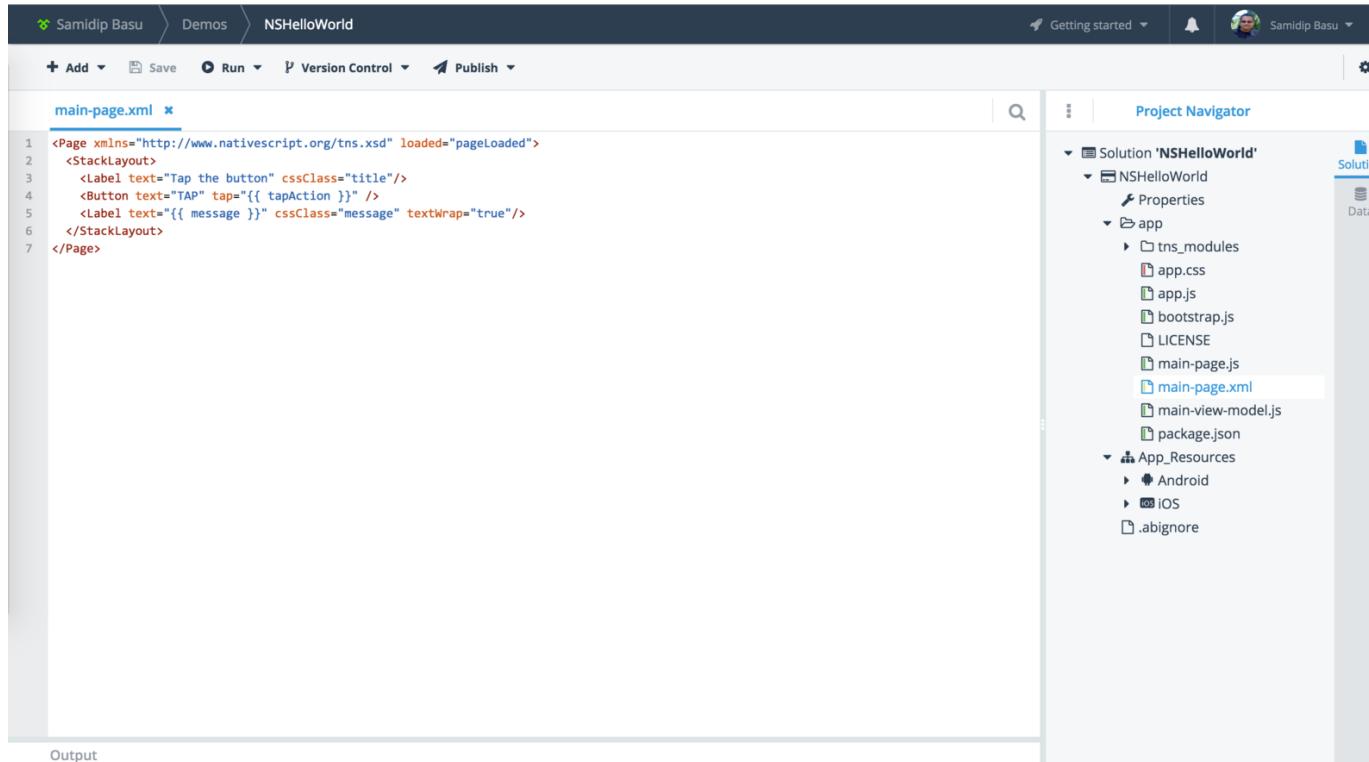
- AppBuilder Hybrid project**
Develop, test, and publish cross-platform hybrid mobile apps using HTML, CSS, and JavaScript.
- AppBuilder Native project**
Develop, test, and build cross-platform native mobile apps using NativeScript with JavaScript or TypeScript. BETA
- AppBuilder Mobile Website project**
Develop, test and export cross-platform mobile websites using HTML, CSS and JavaScript. BETA
- Backend Services project**
Manage a backend for app data and files storage, user management, cloud code, email and push notifications.
- Analytics project**
Discover usage patterns, analyze user data, log exceptions, trace problems and monitor app performance.
- AppFeedback project**
Receive and manage in-app feedback coming directly from your app.
- AppPrototyper project**
Build interactive high-fidelity prototypes for your mobile apps. Export them to AppBuilder. BETA

- NativeScript Blank (JavaScript)**
A blank template for cross-platform application developed with NativeScript.
- NativeScript HelloWorld (JavaScript)**
A sample Hello World cross-platform application developed with NativeScript.
- NativeScript MasterDetail (JavaScript)**
A sample cross-platform application with a master-detail interface developed with NativeScript.
- NativeScript TabbedApp (JavaScript)**
A sample cross-platform application with tab-based navigation developed with NativeScript.
- NativeScript Blank (TypeScript)**
A blank template for a cross-platform application developed with NativeScript using TypeScript.
- NativeScript HelloWorld (TypeScript)**
A sample Hello World cross-platform application developed with NativeScript using TypeScript.
- NativeScript MasterDetail (TypeScript)**
A sample cross-platform application with a master-detail interface developed with NativeScript using TypeScript.
- NativeScript TabbedApp (TypeScript)**
A sample cross-platform NativeScript application with tab-based navigation developed with NativeScript using TypeScript.

You gain Project Templates, Cloud Builds,
Debugging & much more ..

{N} Dev in a Browser

AppBuilder In-Browser Client is rather powerful



The screenshot shows the AppBuilder In-Browser Client interface. The top navigation bar includes 'Samidip Basu', 'Demos', 'NSHelloWorld', 'Getting started', a notification bell, and a user profile. Below the navigation are buttons for '+ Add', 'Save', 'Run', 'Version Control', and 'Publish'. The main area has a code editor titled 'main-page.xml' containing the following XML:

```
1 <Page xmlns="http://www.nativescript.org/tns.xsd" loaded="pageLoaded">
2   <StackLayout>
3     <Label text="Tap the button" cssClass="title"/>
4     <Button text="TAP" tap="{{ tapAction }}" />
5     <Label text="{{ message }}" cssClass="message" textWrap="true"/>
6   </StackLayout>
7 </Page>
```

To the right is the 'Project Navigator' pane, which displays the project structure for 'NSHelloWorld':

- Solution 'NSHelloWorld'
 - Properties
 - app
 - tns_modules
 - app.css
 - bootstrap.js
 - LICENSE
 - main-page.js
 - main-page.xml
 - main-view-model.js
 - package.json
 - App_Resources
 - Android
 - iOS
 - .abignore

Of course, this isn't the only option .. pick your IDE!

Telerik AppBuilder IDEs

{N} truly shines in Developer flexibility

IDE Choices

In-Browser Client



Develop cross-platform mobile apps on the go with a web client that runs in all modern browsers.

Windows Client



This native Windows application provides code navigation, refactoring and real-time on-device updates.

Visual Studio Extension



Enjoy all of the AppBuilder platform's cross-platform capabilities without leaving Visual Studio.

Sublime Text Package



Add a set of commands to build, deploy and LiveSync changes from Sublime Text to your connected devices.

Command-Line Interface



Access the build, deploy and simulator capabilities of AppBuilder from the command line, on Windows or OS X.

Any Platform to suit your Mobile dev needs!



What about UI Composition?

Yes **XML** - it's Shared & X-Platform!

Think of it as lowest common denominator

Show me some UI

{N} uses XAML-like XML

```
<Image source="{thumbnailImageSource}" width="72" height="72"/>
<Label text="{itemTitle}" textWrap="true" cssClass="title" />
<Button text="{authoredBy}" width="150" cssClass="author"/>
```



You build a Visual Tree
Just like XAML/HTML
You get UI elements & Container controls

These become Native UI .. little **Magical!**

Is the UI smart?

{N} supports Rich DataBinding

```
<Image source="{thumbnailImageSource}"  
       width="72"  
       height="72"  
       verticalAlignment="top"/>  
  
<Label text="{{ num_comments ? num_comments + ' comments' : '' }}" />
```



XAML-like Data Binding

Off course .. it's 2-Way

Renders Adaptively for each platform

Data Binding powered by **Polymer** Expressions!

How do I style the UI?

{N} uses the ubiquitous CSS .. hallelujah

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

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



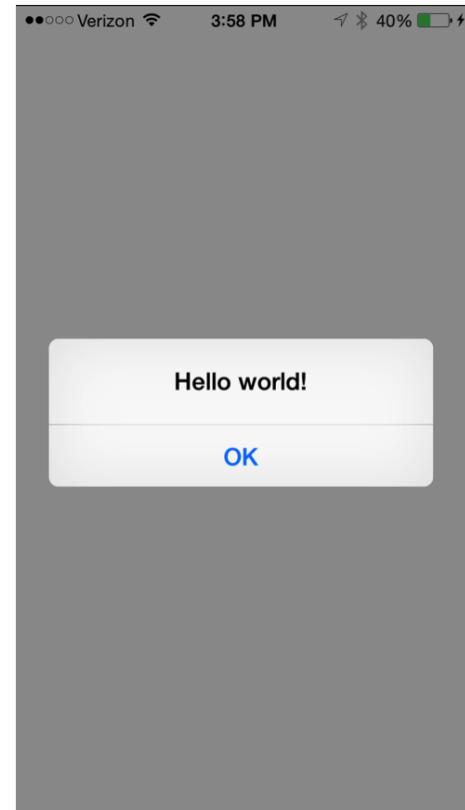
Yep - CSS to style Mobile App elements!

We know you're a **CSS** ninja .. now use it.

Can I invoke Native UI?

{N} supports Native UI element instantiation

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



Native UI

Invoked through JS!



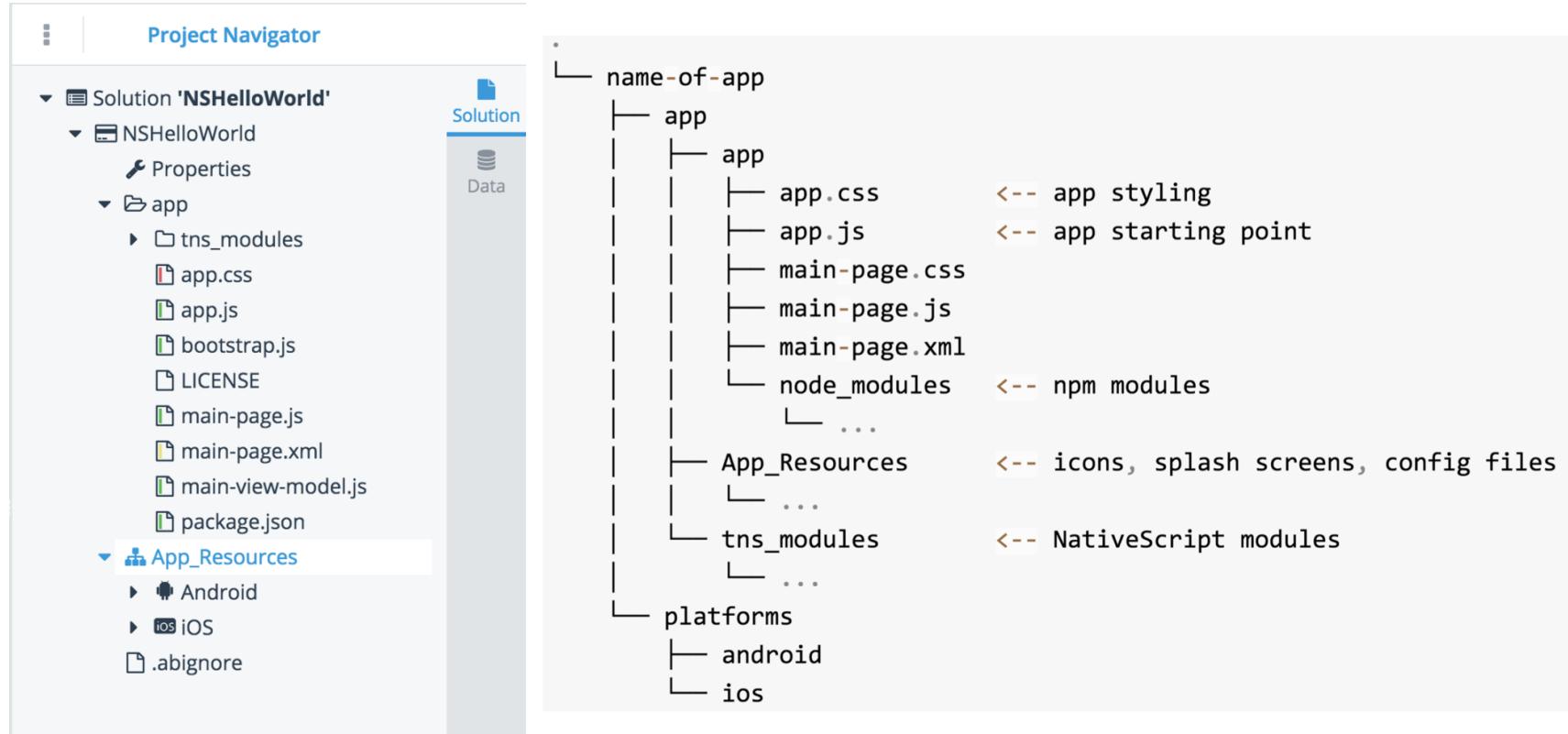
What makes up a {N} Project?

You're asking all the right questions ..

Combination of UI, Logic, Styling & **Modules!**

A Typical {N} Project

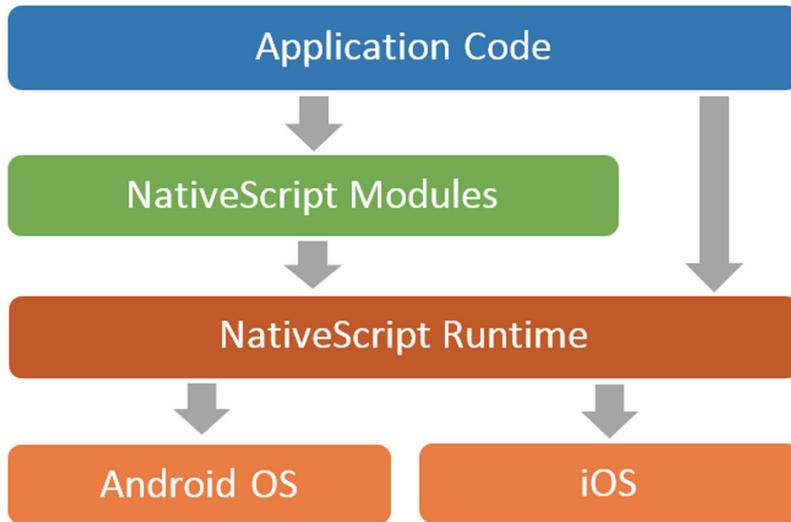
All your components in one place



They say a picture is worth a 1000 words ..

What's a Module?

{N} bridge that takes functionality X-Platform



- Just like a Node module
- Dozens available | You can write custom modules

Show a Module in action

{N} allows you to bring in what's needed

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

```

- This is a generic HTTP Module
- Works the same way in each Platform

Module to Native

{N} does the mapping .. so you don't have to

```
var fileSystem = require("file-system");
new fileSystem.File(somePath);
```



Translates this as ..

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



Translates this as ..

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



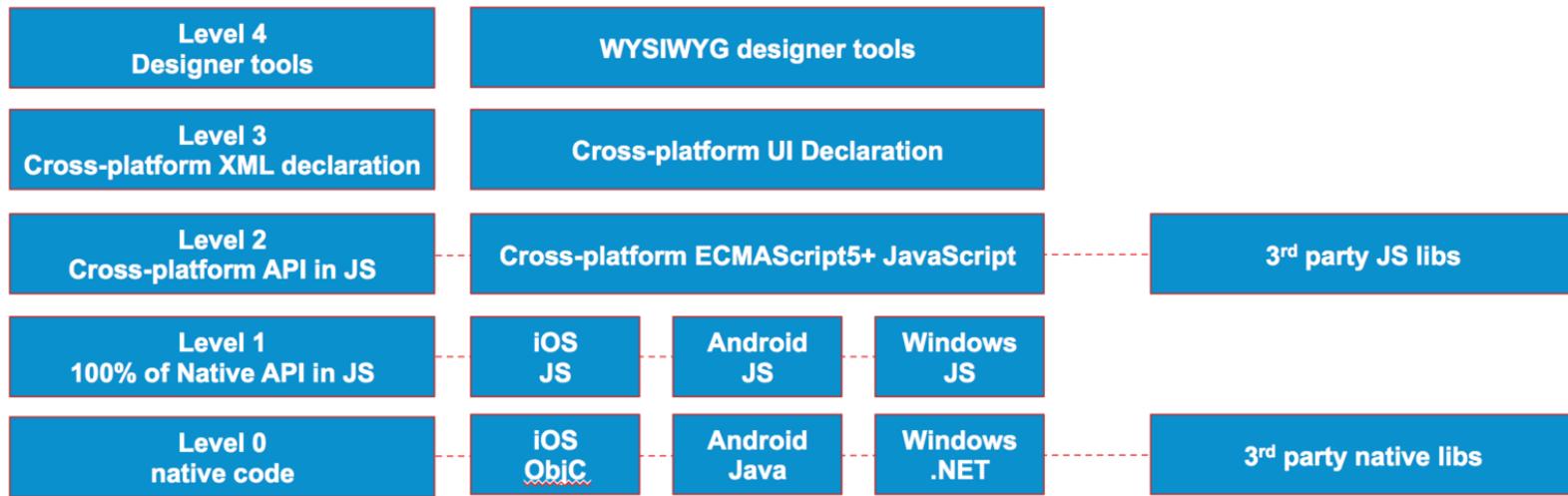
You had me at Native with JS!

Yeah, we're just as **excited**.

There is a lot we covered though ..

Gimme a 10K Feet View

Here you go .. the layers of {N} magic!



{N} also sports **integrated support** with Angular 2!



Reuse your Web & TypeScript skills ..

Why {N} again?

Reasons You'll Love it



Truly Native UX

Create the performance and experience of a truly native app with a fraction of the development effort.



Use 3rd Party JavaScript Libraries

Reuse all the available JavaScript libraries that do not have browser or other platform dependencies.



Same day support for new platforms

Get same day support for the latest updates to the native platforms.



XML UI declaration

Declare app UI in simple, easy to read cross-platform XML-based format.



Open Source

The NativeScript framework is open source under the Apache 2.0 license and the source code is available on GitHub.



Rich data-binding

Use data binding in the XML declaration with powerful data binding syntax engine.



You're a Mobile Ninja

{N} is here to help!



Feel the Native Zen

Tooling that elevates X-Plat Mobile



[Getting Started](#)



[{N} with Telerik Platform](#)



[App Showcase](#)



[Solid Docs | Forums](#)



[Contribute Back](#)

Go build your dream app!



By



Native X-Platform Mobile with JS
[NativeScript Blog](#) | [@NativeScript](#)