NativeScript.org







# Diving Deep

## Summer of NativeScript

TJ VanToll, Clark Sell

# So far you've learned

- What NativeScript is
- How to build apps with NativeScript
- How to use UI controls, use NativeScript modules, and add application logic

# Today

- Accessing native code
- Debugging
- Using npm modules
- NativeScript plugins

# Today

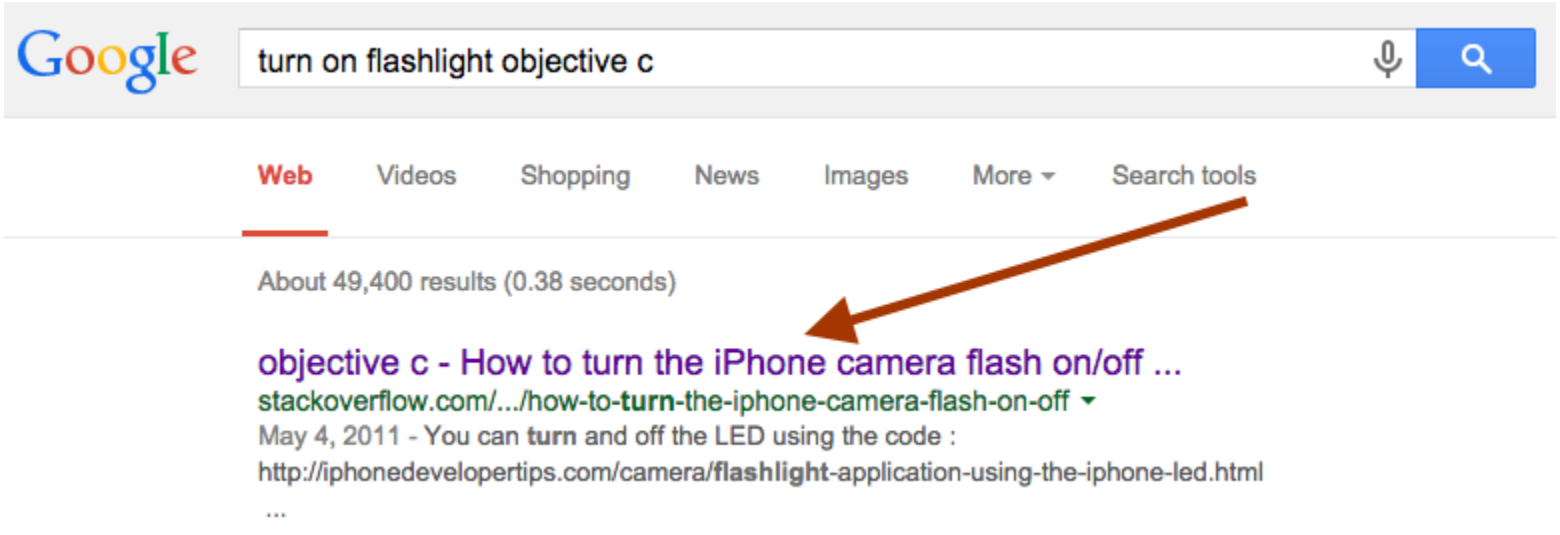
- Accessing native code
- Debugging
- Using npm modules
- NativeScript plugins

# Accessing native code

All iOS and Android APIs are available to your NativeScript app in JavaScript directly.

# Example

Suppose you want to add a flashlight to your NativeScript-written iOS app.


Google
turn on flashlight objective c
Web
Videos
Shopping
News
Images
More
Search tools
About 49,400 results (0.38 seconds)
objective c - How to turn the iPhone camera flash on/off ...
stackoverflow.com/.../how-to-turn-the-iphone-camera-flash-on-off
May 4, 2011 - You can turn and off the LED using the code :
http://iphonedevelopertips.com/camera/flashlight-application-using-the-iphone-led.html
...

53

```
#import <AVFoundation/AVFoundation.h>
```

...

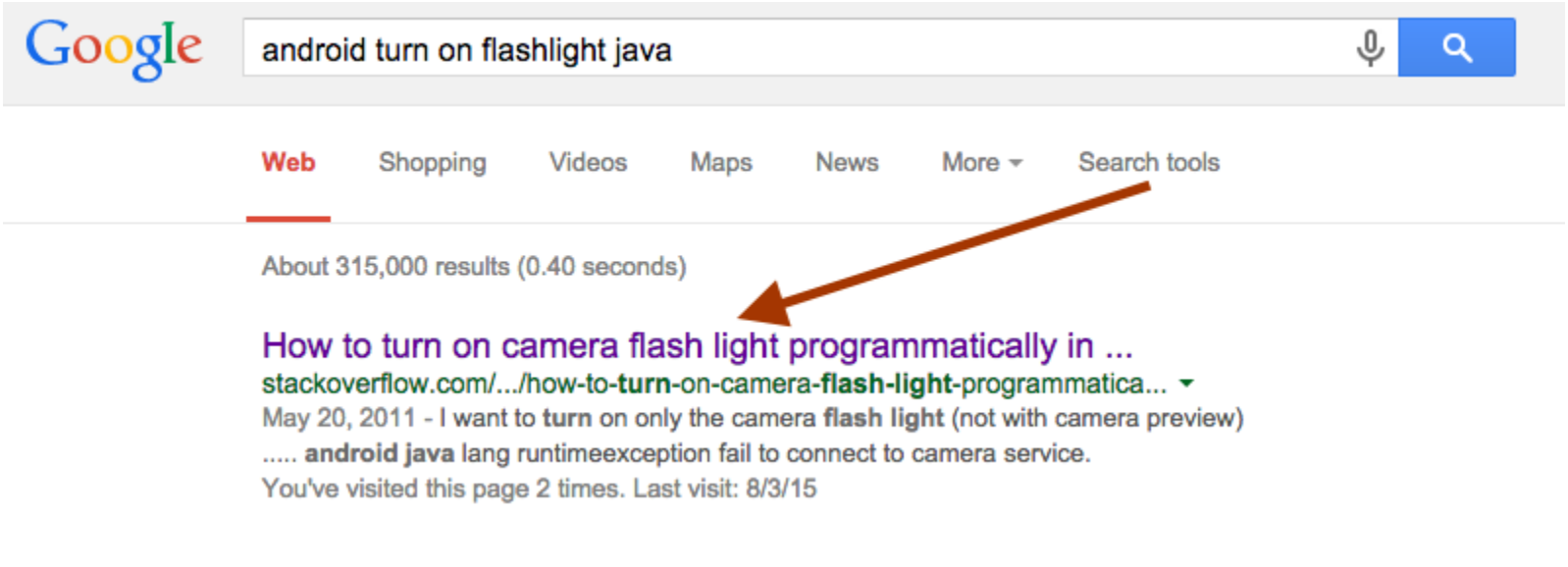
```
- (void) turnTorchOn: (bool) on {

    // check if flashlight available
    Class captureDeviceClass = NSClassFromString(@"AVCaptureDevice");
    if (captureDeviceClass != nil) {
        AVCaptureDevice *device = [AVCaptureDevice defaultDeviceWithMediaType:AVMediaTypeVic
        if ([device hasTorch] && [device hasFlash]){

            [device lockForConfiguration:nil];
            if (on) {
                [device setTorchMode:AVCaptureTorchModeOn];
                [device setFlashMode:AVCaptureFlashModeOn];
                //torchIsOn = YES; //define as a variable/property if you need to know statu
            } else {
                [device setTorchMode:AVCaptureTorchModeOff];
                [device setFlashMode:AVCaptureFlashModeOff];
                //torchIsOn = NO;
            }
            [device unlockForConfiguration];
        }
    } }
```

# The same function in NativeScript

```
function turnTorchOn(on) {
    var device = AVCaptureDevice.defaultDeviceWithMediaType(AVMediaTypeVideo);
    if (!!device) { return; }
    device.lockForConfiguration(null);
    device.torchMode = on ? AVCaptureTorchMode.AVCaptureTorchModeOn :
      AVCaptureTorchMode.AVCaptureTorchModeOff;
    device.flashMode = on ? AVCaptureFlashMode.AVCaptureFlashModeOn :
      AVCaptureFlashMode.AVCaptureFlashModeOff;
    device.unlockForConfiguration();
}
```


Google
android turn on flashlight java
Web
Shopping
Videos
Maps
News
More
Search tools
About 315,000 results (0.40 seconds)
How to turn on camera flash light programmatically in ...
stackoverflow.com/.../how-to-turn-on-camera-flash-light-programmatica...
May 20, 2011 - I want to turn on only the camera flash light (not with camera preview)
..... android java lang runtimeexception fail to connect to camera service.
You've visited this page 2 times. Last visit: 8/3/15

252

For **Checking availability of flash** in device:

You can use the following

```
context.getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA_FLASH);
```

which will return true if a flash is available, false if not.

Code Snippet to turn on camera flash light.

```
Camera cam = Camera.open();
Parameters p = cam.getParameters();
p.setFlashMode(Parameters.FLASH_MODE_TORCH);
cam.setParameters(p);
cam.startPreview();
```

Code snippet to turn off camera led light.

```
  cam.stopPreview();
  cam.release();
```

# The same code in NativeScript

```
var packageManager = application.android.currentContext.getPackageManager();
if (!packageManager.hasSystemFeature(
  android.content.pm.PackageManager.FEATURE_CAMERA_FLASH) {
    return;
}

var camera = android.hardware.Camera.open(0);
var p = camera.getParameters();
p.setFlashMode(camera.Parameters.FLASH_MODE_TORCH);
camera.setParameters(p);
camera.startPreview();

// Turning the camera off
camera.stopPreview();
camera.release();
```

# Converting native to JS, a few tips

- Check the docs when you hit an advanced scenario
  - Android:
    - http://docs.nativescript.org/runtimes/android/overview
  - iOS:
    - http://docs.nativescript.org/runtimes/ios/Overview

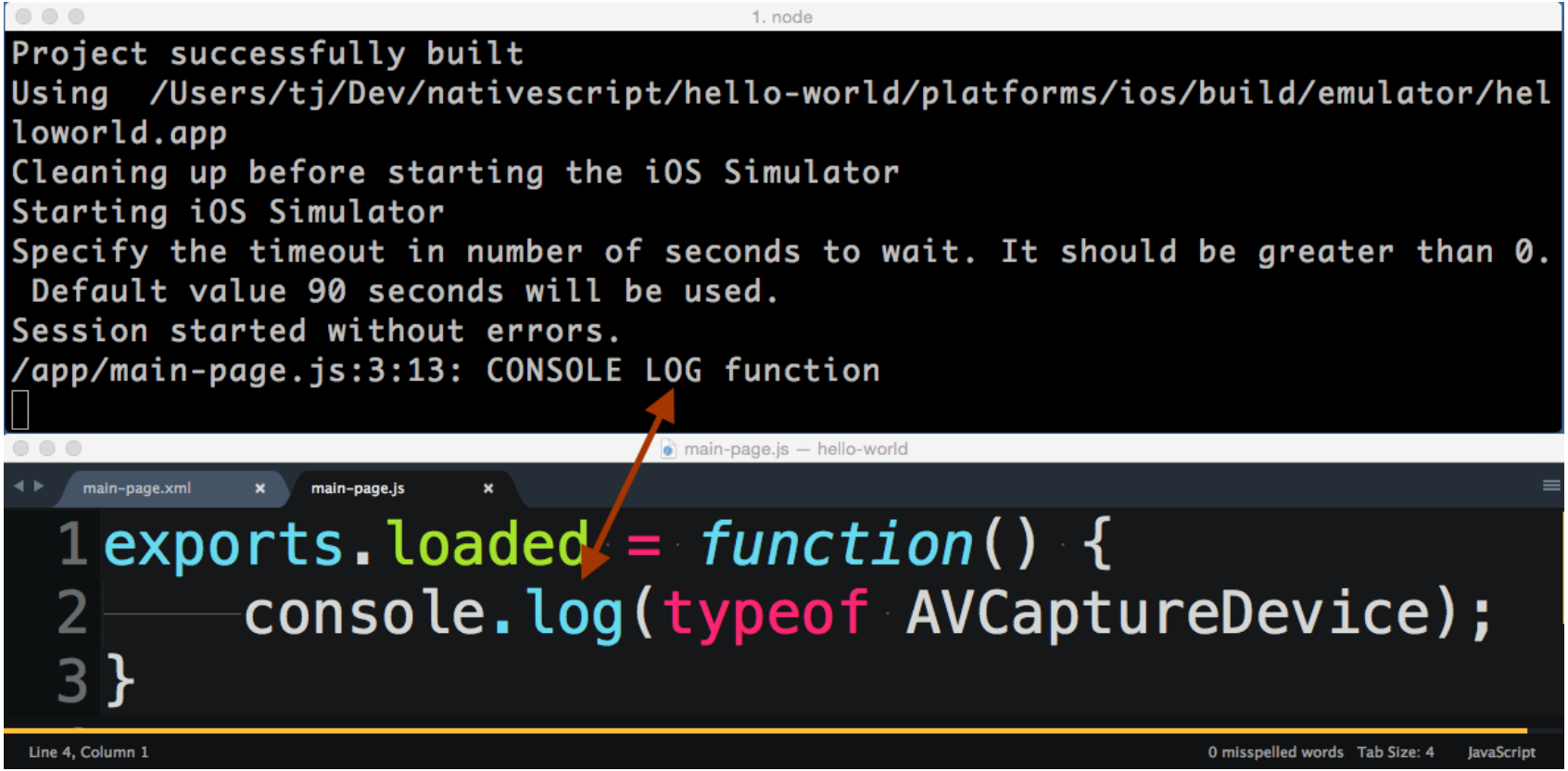
- Use NativeScript's TypeScript definition files as a reference for native APIs
  - Android
    - https://raw.githubusercontent.com/NativeScript/NativeScript/master/android17.d.ts
  - iOS
    - https://raw.githubusercontent.com/NativeScript/NativeScript/master/ios.d.ts
- Try running native APIs with the debug tooling.

# Today

- Accessing native code
- Debugging
- Using npm modules
- NativeScript plugins

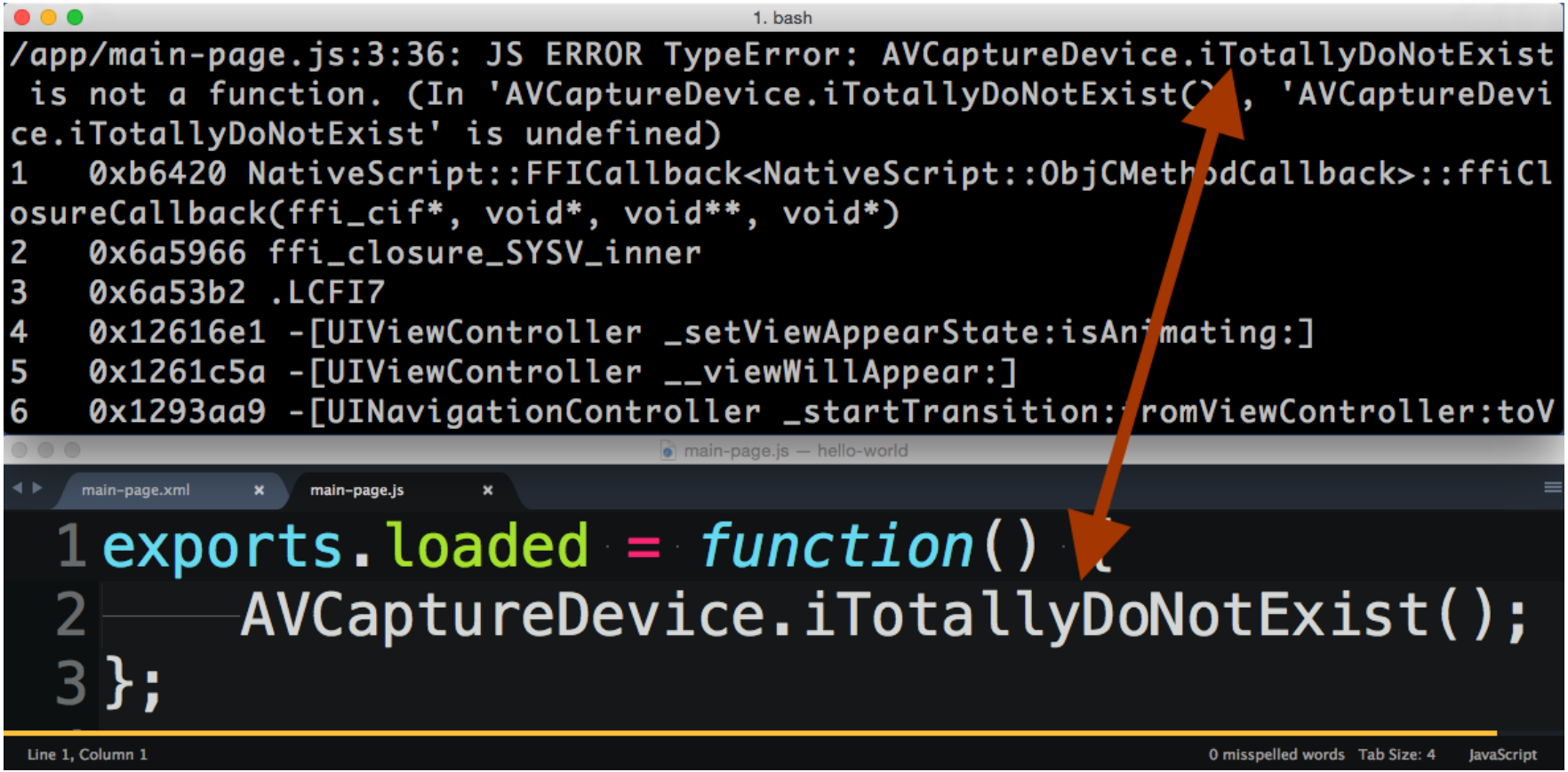
# Console logging

- The NativeScript CLI automatically sends the result of all console.log calls to the terminal.


1. node
Project successfully built
Using /Users/tj/Dev/nativescript/hello-world/platforms/ios/build/emulator/helloworld.app
Cleaning up before starting the iOS Simulator
Starting iOS Simulator
Specify the timeout in number of seconds to wait. It should be greater than 0. Default value 90 seconds will be used.
Session started without errors.
/app/main-page.js:3:13: CONSOLE LOG function
main-page.js — hello-world
main-page.xml
main-page.js
1 exports.loaded = function() {
2 console.log(typeof AVCaptureDevice);
3 }
Line 4, Column 1
0 misspelled words
Tab Size: 4
JavaScript

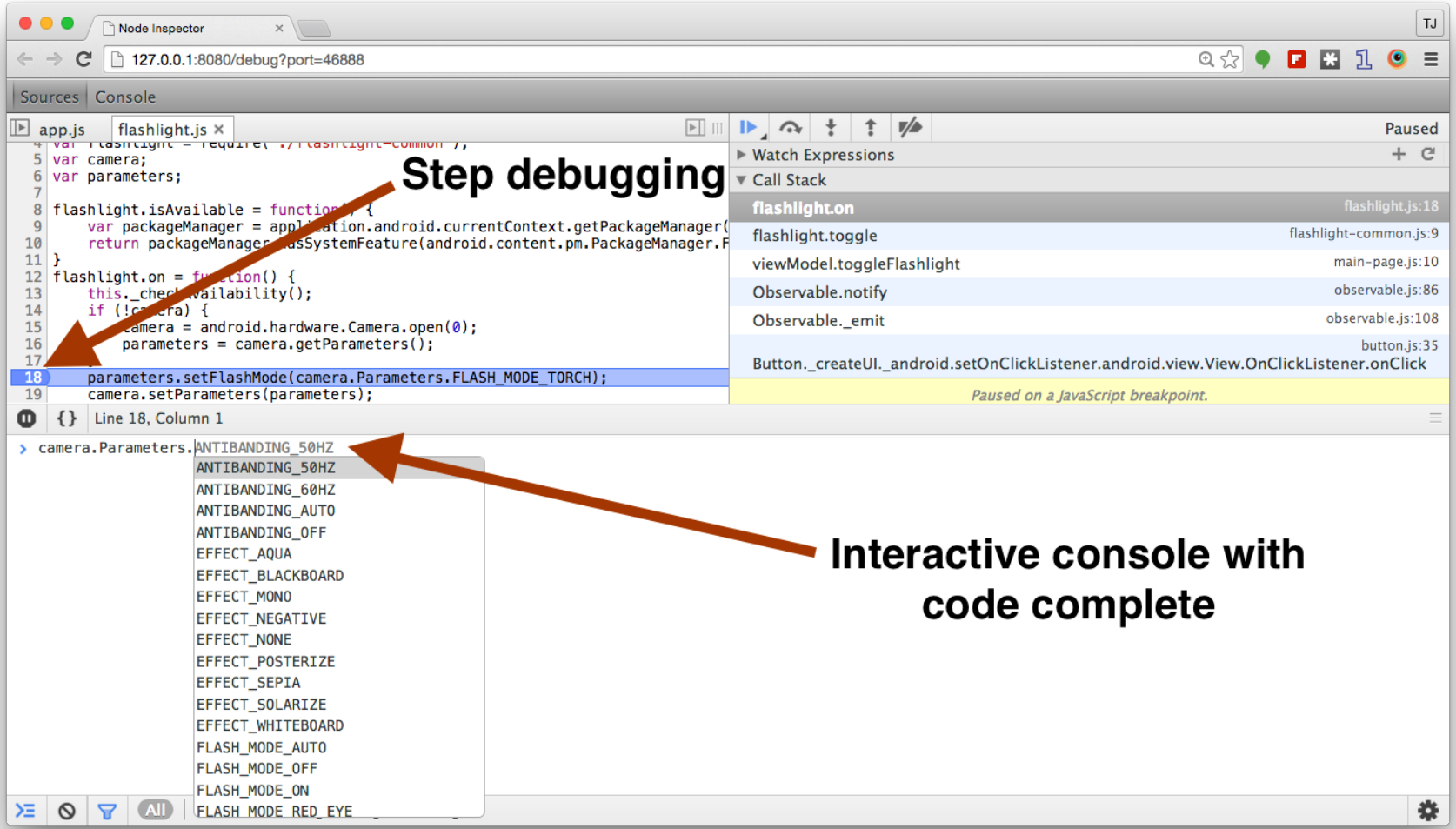
# Stack traces

- The NativeScript CLI also sends stack traces to the terminal when things go wrong.


1. bash
/app/main-page.js:3:36: JS ERROR TypeError: AVCaptureDevice.iTotallyDoNotExist
is not a function. (In 'AVCaptureDevice.iTotallyDoNotExist(), 'AVCaptureDevi
ce.iTotallyDoNotExist' is undefined)
1 0xb6420 NativeScript::FFICallback<NativeScript::ObjCMethodCallback>::ffiCl
osureCallback(ffi_cif*, void*, void**, void*)
2 0x6a5966 ffi_closure_SYSV_inner
3 0x6a53b2 .LCFI7
4 0x12616e1 -[UIViewController _setViewAppearState:isAnimating:]
5 0x1261c5a -[UIViewController __viewWillAppear:]
6 0x1293aa9 -[UINavigationController _startTransition:fromViewController:toV
main-page.js — hello-world
main-page.xml
main-page.js
1 exports.loaded = function() {
2 AVCaptureDevice.iTotallyDoNotExist();
3 };
Line 1, Column 1
0 misspelled words
Tab Size: 4
JavaScript

# Web inspector

- Use the `tns debug` command to launch.
- Allows for step debugging, an interactive console and more.
- Visual tree debugging and performance benchmarking is coming soon.


Node Inspector
127.0.0.1:8080/debug?port=46888
Sources
Console
app.js
flashlight.js
Step debugging
var camera;
var parameters;
flashlight.isAvailable = function() {
var packageManager = application.android.currentContext.getPackageManager(
return packageManager.hasSystemFeature(android.content.pm.PackageManager.F
}
flashlight.on = function() {
this._checkAvailability();
if (!camera) {
camera = android.hardware.Camera.open(0);
parameters = camera.getParameters();
parameters.setFlashMode(camera.Parameters.FLASH_MODE_TORCH);
camera.setParameters(parameters);
Paused
Watch Expressions
Call Stack
flashlight.on
flashlight.js:18
flashlight.toggle
flashlight-common.js:9
viewModel.toggleFlashlight
main-page.js:10
Observable.notify
observable.js:86
Observable._emit
observable.js:108
button.js:35
Button._createUI._android.setOnClickListener.android.view.View.OnClickListener.onClick
Paused on a JavaScript breakpoint.
Line 18, Column 1
camera.Parameters.ANTIBANDING_50HZ
ANTIBANDING_50HZ
ANTIBANDING_60HZ
ANTIBANDING_AUTO
ANTIBANDING_OFF
EFFECT_AQUA
EFFECT_BLACKBOARD
EFFECT_MONO
EFFECT_NEGATIVE
EFFECT_NONE
EFFECT_POSTERIZE
EFFECT_SEPIA
EFFECT_SOLARIZE
EFFECT_WHITEBOARD
FLASH_MODE_AUTO
FLASH_MODE_OFF
FLASH_MODE_ON
FLASH_MODE_RED_EYE
All
Interactive console with code complete

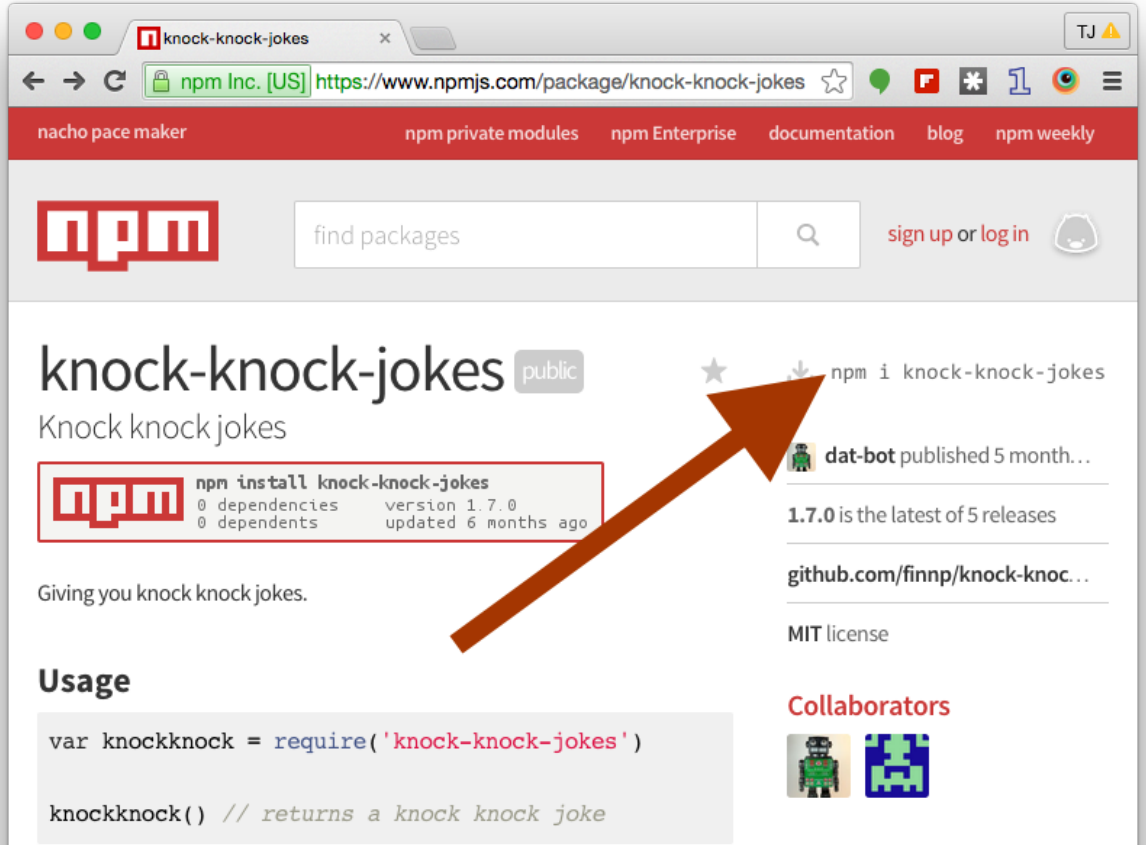
# Today

- Accessing native code
- Debugging
- Using npm modules
- NativeScript plugins

# Using npm modules

- NativeScript modules adhere to the same CommonJS spec Node modules do.
- Installing modules from npm works just like it does in Node apps.

# Example: Knock knock jokes



```
$ npm install knock-knock-jokes --save
```

- Notes
  - Make sure to run the install in the root of the app.
  - The --save flag tells npm to save the dependency in your app's package.json file.
  - At build time the NativeScript CLI copies the appropriate files from node_modules into the app itself.

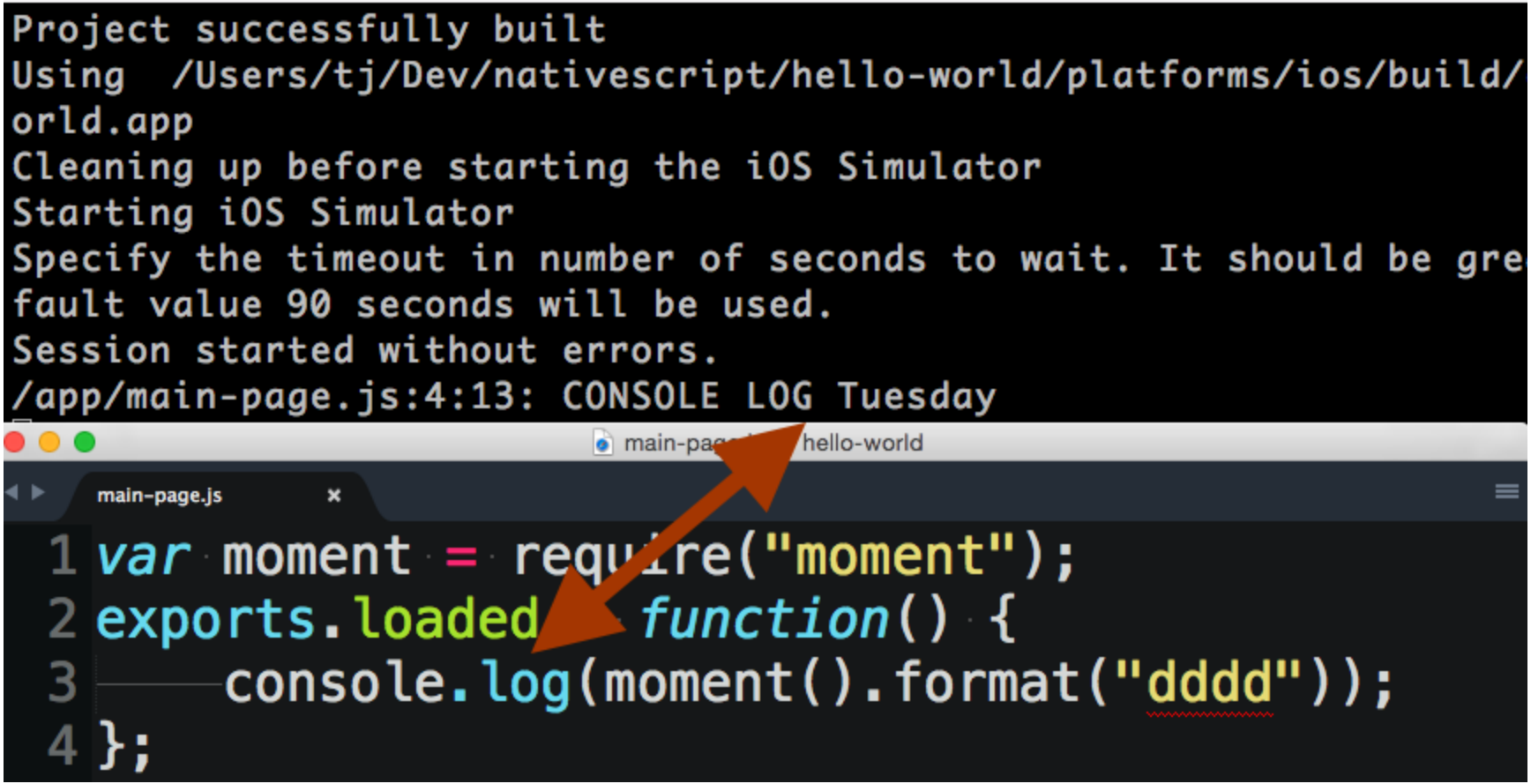
# Using knock knock jokes

```
var knockknock = require("knock-knock-jokes");
console.log(knockknock());
```

- It’s that easy!
- Let’s look a slightly more practical example.

# Using Moment.js

```
$ npm install moment --save
```


Project successfully built
Using /Users/tj/Dev/nativescript/hello-world/platforms/ios/build/
orld.app
Cleaning up before starting the iOS Simulator
Starting iOS Simulator
Specify the timeout in number of seconds to wait. It should be gre
fault value 90 seconds will be used.
Session started without errors.
/app/main-page.js:4:13: CONSOLE LOG Tuesday
hello-world
main-page.js
1 var moment = require("moment");
2 exports.loaded = function() {
3 console.log(moment().format("dddd"));
4 };

# Supported npm modules

- Most npm modules will work in {N} apps.
- Notable ones that won’t are…
  - Modules that depend on browser APIs such as the DOM (e.g. jQuery)
  - Modules that depend on Node.js APIs not present in NativeScript

# Today

- Accessing native code
- Debugging
- Using npm modules
- NativeScript plugins

# NativeScript plugins

- {N} plugins are also npm modules.
- However, {N} plugins have the added ability to run native code, as well as use native iOS and Android SDKs.

# Installing plugins

- The following installs the NativeScript flashlight plugin:

```
$ tns plugin add nativescript-flashlight
```

# Plugins

- `tns plugin add` does the following
  - Installs the plugin from npm (including saving dependency in your package.json)
  - Manages Android and iOS configuration files.
  - Installs any necessary iOS and Android SDKs that the plugin needs.

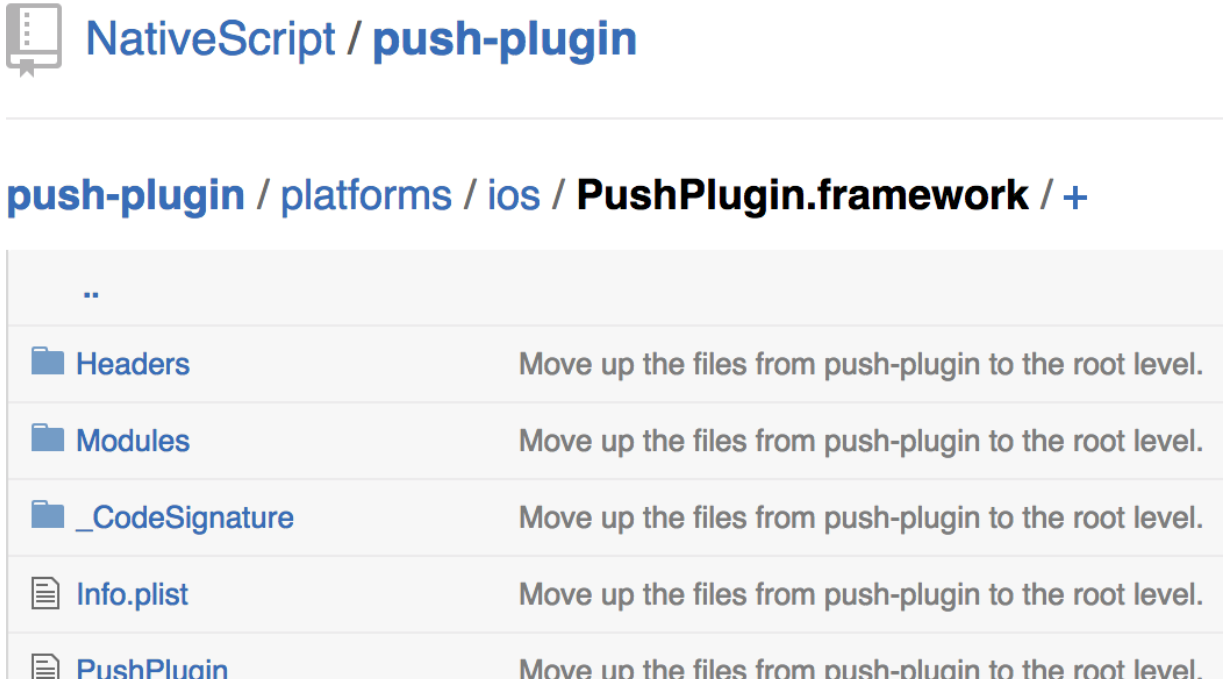
- For example, using the camera on Android requires camera permissions, but the plugin install takes care of that.

**nativescript-flashlight** / platforms / android / **AndroidManifest.xml**

5 lines (5 sloc) | 0.245 kB

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android">
3      <uses-permission android:name="android.permission.CAMERA" />
4      <uses-feature android:name="android.hardware.camera" />
5  </manifest>
```

- The NativeScript push plugin uses iOS . framework files, but again, the plugin install takes care of that.



- After installing, using {N} plugins is just like using an npm module.

```
var flashlight = require("nativescript-flashlight");
flashlight.on();
```

# Where to find NativeScript plugins?

- npm
  - https://www.npmjs.com/search?q=nativescript

# Verified Plugin Marketplace


- http://plugins.telerik.com/

# Questions?

- https://groups.google.com/forum/#!forum/nativescript
- https://twitter.com/nativescript

NativeScript.org

# Thanks!