Typescript, Angular, 和移动端的跨平台开发

Ryan Chen 陈亮, Engineering Manager
Chromecast and Home, Google

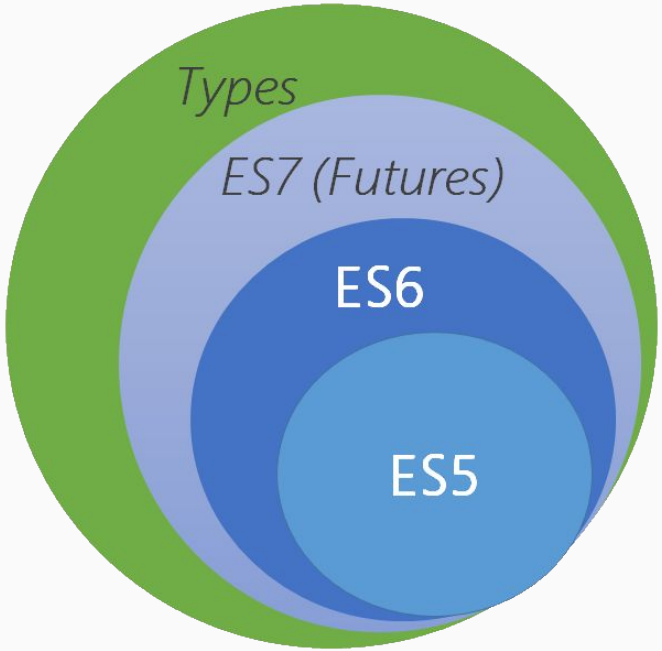

TS

A

ionic

N

What’s TypeScript?

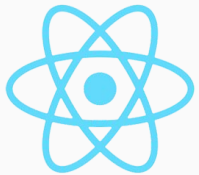


What’s TypeScript?

- A **statically typed superset** of JavaScript that **compiles** into plain JavaScript.
- **Future** features + Typing + Tooling (Compiler / IDE)
- Released in 2012 and open source.
- Anders Hejlsberg: Typescript makes JavaScript **scale**.
  - Better readability and integratable software.
  - Any browser, host, and OS.

TypeScript Type System

- TypeScript compiler (tsc)
  - **tsconfig.json** - configures compilation options for JS
  - Static type checking and code refactoring.
- Types are optional and can be inferred by tsc.
- Works with all the popular frameworks:



TypeScript Features

- Enums 枚举
- Interfaces 介面
- Decorators 装饰器
- Protected / Private
- Union Types
- Modules (ES6 模块)
- Async / Await (generator)
- Exponential operator (**)
- keyof
- Object Spread
- Mixin 混入
- Iterator

TypeScript Interfaces

- Interfaces describe the shape of your JS objects.
- Interfaces disappear during compilation to JS.
- Coercions do not add runtime checks.

```
interface MediaInformation {
  title: string,
  description: string,
  type: CustomMediaType,
  url: Url
}
```

TypeScript Decorators

- **Decorators** add annotation and meta-programming syntax for
  - Classes **@Component @Injectable**
  - Properties **@Input @Ouput @ContentChildren**
- Observe, modify, or replace existing objects to allow additional functionalities.

```
@Component({
  selector: 'tab-group',
  templateUrl: 'tab-group.component.html',
  changeDetection: OnPush
})
class TabGroup {
  @ContentChildren(Tab) tabs: QueryList<Tab>;
  @ViewChildren(Header) headers: QueryList<Header>;
}
```

TypeScript Typing

- **--strictNullChecks** prevents common mistakes.
- Type assertions <expr>**!.**<method>
- Type casting <expr> **as** <type>
- **any** describes a type of variables that we do not know at compile time.
  - Allow integration with 3rd party JS Library
  - Opt-in/out type checking flexibility
- **never** catches functions that throw error.

TypeScript External Types

- Type declarations files (.d.ts): Interfaces, enums etc.
- Existing browser and DOM types: lib.es6.d.ts
- DefinitelyTyped - 3000+ common .d.ts files
- Editors (i.e. VS Code) can understand npm modules that have type definitions.

TypeScript Adoptions

- Google - All Angular products by default.
  - Google Express, Firebase, Analytics, Cloud, Finance, etc.
- Microsoft - Visual Studio Code
- Netflix - Many web browser products (i.e. video player) and internal services.
- Palantir, Slack
- Finance: Accenture, Forbes, Capital One, etc.
- Entertainment: NBA, Royal Caribbean, ...

(Angular) => return Platform

Why Angular?

- **Speed and Performance**
  - Complete Rewrite from AngularJS with 5x improvement.
  - Angular 4 : Rewrote View Engine => 40% smaller size application.
  - Laze-load.
- **Cross Platform for Mobile:** DOM Decoupling.
- **Platform** not just a framework => High quality end to end experiences.
- **Scalability, trustworthy, community.**
  - Built to scale.
  - One change in Angular => thousands of tests run as a result. (Google and outside)
  - No breaking change in incremental versions.

Why Angular?

CORE
核心

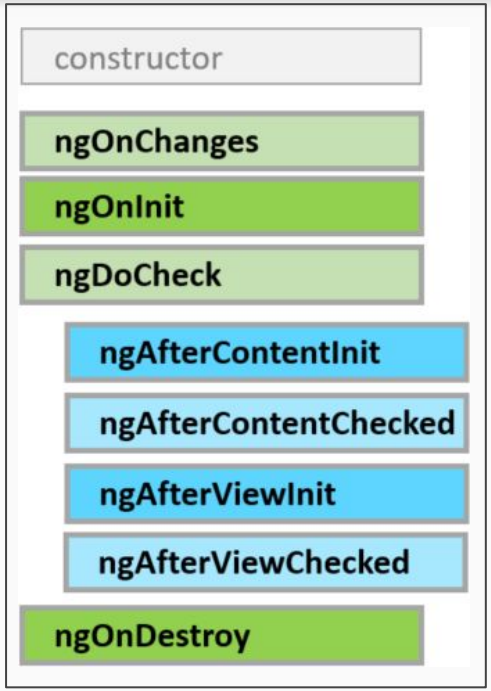
LIBRARIES
库

TOOLS
工具

Angular Core

- Expressive declarative **components** and **directives**.
  - Cross-Platform: Decoupled rendering from the DOM (NativeScript, Universal)
  - Content Projection
  - Pipes
- Dependency Injection
- Zones
  - Execution context for hooking onto async tasks.
  - Change Detection strategies.

Angular Lifecycle Hooks



- A component or directive has a lifecycle managed by Angular.
- Angular creates it, renders it, creates and renders its children, checks it when its data-bound properties change, and destroys it before removing it from the DOM.
- Angular offers lifecycle hooks that provide visibility into these key life moments and the ability to act when they occur.

Performance Myth

Performance result from js-benchmark-framework:

Framework	**vue-2.3.3**	**angular-4.1.2**	**react-15.5.4**
Geometric Mean	1.07	1.12	1.17

- 10 types of operations, fastest:
  - Angular fastest in 6: Replacing all rows; partial update; select row; swap rows; remove rows; append row to large table.
  - Vue fastest in 4: create rows, create many rows, clear many rows, startup time.
  - React: 0.

Is that the fair truth?

Angular Change Detection

- **NgZone** - Creates an execution context where Angular monkey patches the asynchronous calls and emit **onTurnDone** event.
- **ApplicationRef** - Triggerred by **onTurnDone** and checks the whole component tree for change detections.
- **ChangeDetectorRef** - checks only the current component and its children.

```
ObservableWrapper.subscribe(
  this.zone.onTurnDone, () => {
    this.zone.run(() => { this.tick(); });
  });

tick() {
  this.changeDetectorRefs.forEach((detector)
=> {
    detector.detectChanges();
  });
}
```

Angular Change Detection

- Every component has a change detector.
- Change detector propagates bindings from **top to leaves**.
  - Directed tree: More performant, predictable, and debuggable than cycles.
- Hundred of thousands of checks per few milliseconds.
  - Done per changeable event (DOM event, XHR, setTimeout/Interval) because JS **objects are mutable**.

To the rescue...

Immutability and Observable

Observable

- Observable (by Ben Lesh, speaker from last JS Conf China.)
  - A stream of data to represent any number of things over any amount of time.
  - Lazy, cancellable, and don’t generate values until someone subscribes to it.
- Angular Uses RxJs Observable for subscription based processing of asynchronous operations.

Observable

```
let observable = new Observable(observer => {
  const operation = fn((value, err) => {
    if (!err) {
      observer.next(val);
      if (<condition meets for last value>) { observer.complete(); }
    } else {
      observer.error(err);
    }
  });
  return () => { cancelOp(operation); }
});
```

Angular Change Detection OnPush

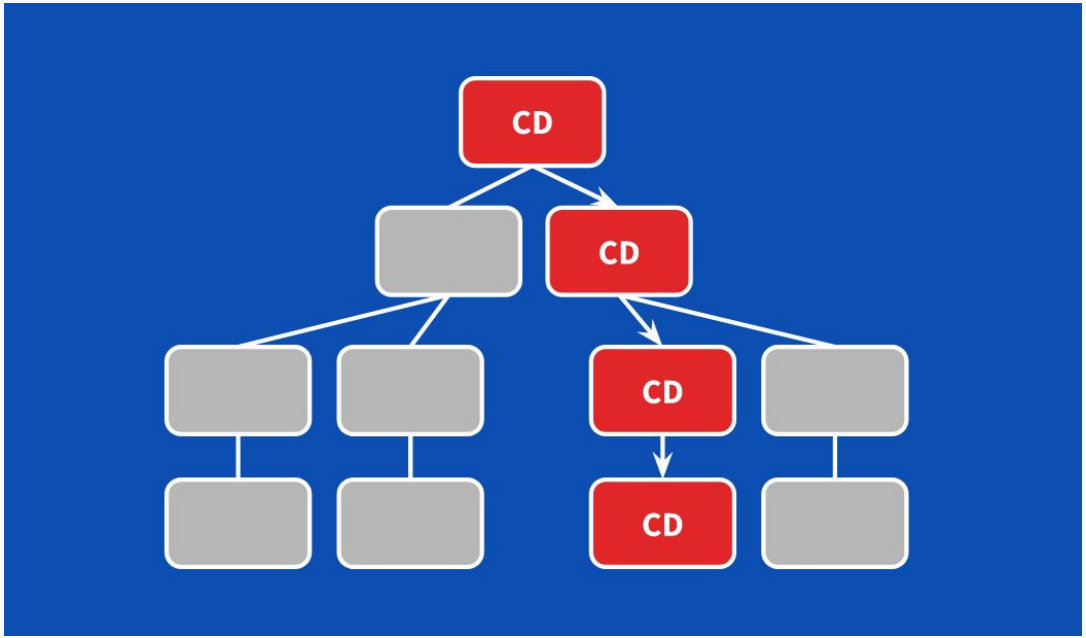
- Immutable input
  - Component can only change **if its input properties change**.
  - Data of component's parent and all the way to the root has to change.
- Observable input
  - Component can only change **if one of its input emits an event**.
  - Component's data can change without its input or parent or up change.

Angular Async Pipe and Observable

- Subscribes to an observable and returns latest value emitted.
- New value => marks the component to be checked for changes.
- Component Destroyed? Automatically unsubscribe to prevent memory leak.

```
@Component({
  selector: 'current-time',
  template: 'Current Time: {{ timeObservable | async }}</div>'
})
export class CurrentTimeComponent {
  timeObservable = new Observable<string>((observer: Subscriber<string>) => {
    setInterval(() => observer.next(new Date().toString()), 1000);
  });
}
```

Angular Change Detection



- Default change detection complexity: **O(n)**
- OnPush Observable complexity: **O(log(n))**

Angular Libraries

- Forms - Validations and Two-way data binding
- Router
- Animations
- Material Components
- i18n

Angular Router Config

```
import {RouterModule, Routes} from
'@angular/router';

const routes: Routes = [
  {path: '', component: WatchListComponent},
  {path: 'videos/:id', component:
VideoDetailsComponent},
  {path: 'index', redirectTo: '/', pathMatch:
'full'},
]
```

```
<a routerLink="/index">Index</a>
<a routerLink="/video">Video</a>
<router-outlet></router-outlet>
```

Angular Dynamic Router

Dynamic Router - Automatic code-splitting to load code required to render the view they request.

- Default: Register the module only when needed.
- Webpack implementation creates bundle splitting and serve file url.

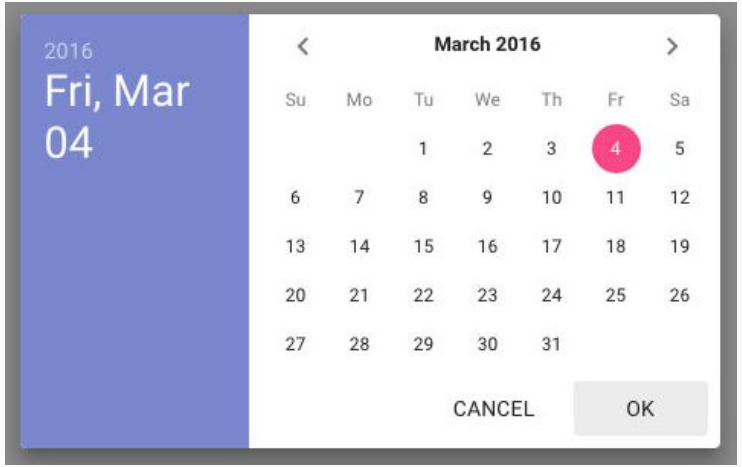
```
import { Routes } from '@angular/router';

export const ROUTES: Routes = [
  { path: '',       component: HomeComponent },
  { path: 'detail', loadChildren: './path/to/module#ModuleName'}
];
```

Angular Animation

- Built on top of standard Web Animation API and run natively on browsers that supports it.
- Part of Angular 2 core originally but now separated into @angular/platformBrowser/animation.
  - Smaller code size.
  - Better adaptations to other platforms like native.

Angular Material

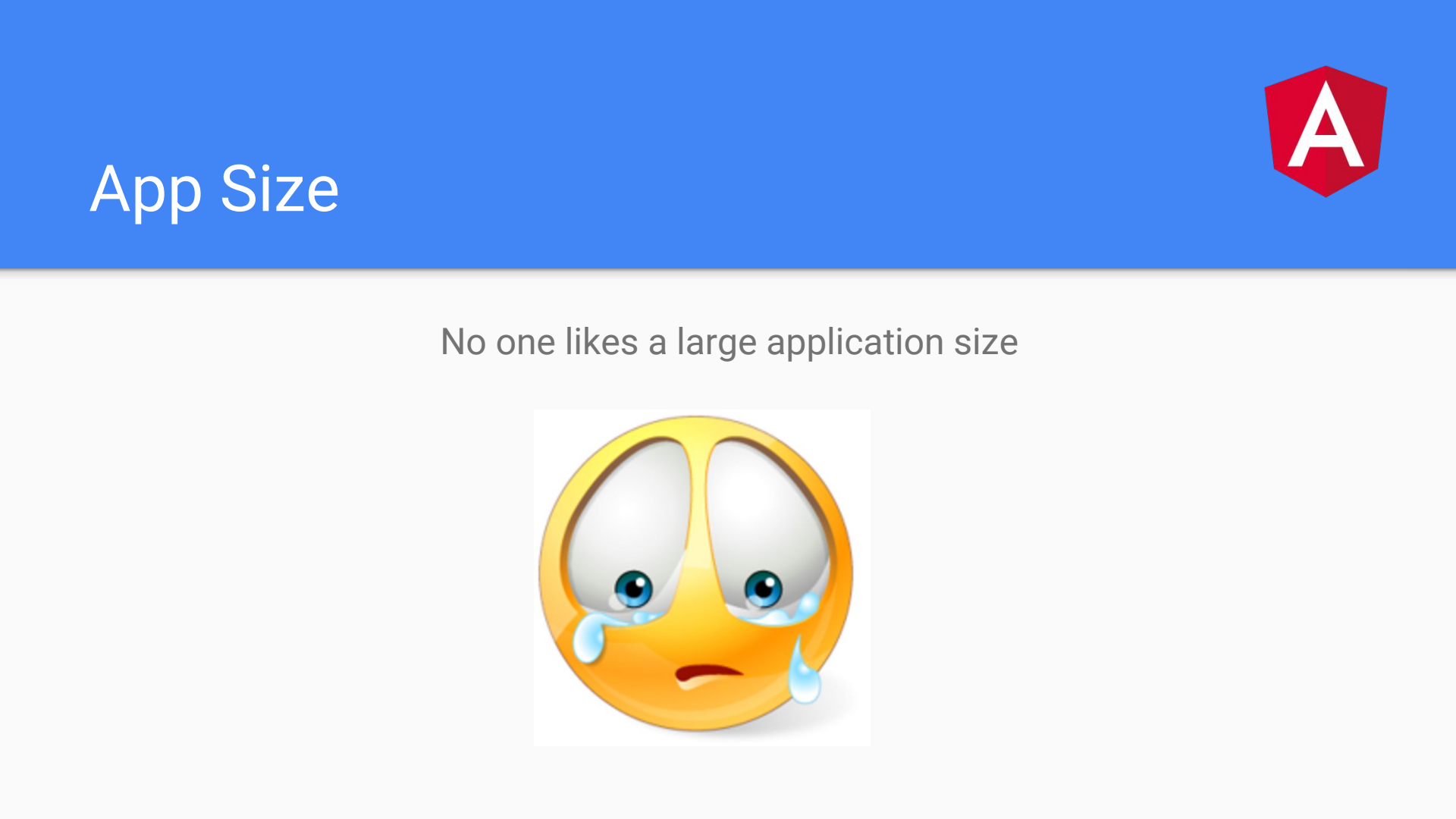
- Mordern UI components work across Mobile, Web, and Desktop.
  - Form: Date Picker, AutoComplete, Slider...
  - Navigation: Menu, Toolbar...
  - Layout: Tabs, Grids, Lists…
  - Controls: Progress spinner and bar…
  - DataGrid
- Themeable and Internationalizable.
- Tuned for performance.



Angular Tooling

- Ahead of Time (AOT) Compiler
- Angular Universal
- Command Line Interface (Angular CLI)
- Augury (Chrome Dev Tool)
- Language Services: IDE Integration
- Protractor and Karma for testing.

App Size



No one likes a large application size

Angular AOT

- Ahead-of-time compilation: Runs once at build time.
- Eliminates the need to package the Angular compiler (half the size of Angular library) to your production application.
- Detect template errors early.
- Used by Mobile Frameworks: Ionic & NativeScript.
- Tree Shaking: Remove any dead code not used in final bundle by WebPack.

Angular Universal

- Pre-compiles the app into an HTML/JS/CSS offline in a build step.
- Host on CDN for caching.
  - First time users instantly see result!
  - Optimized for search engine.
- Preboot creates hidden div and records user interactions.
- Angular starts boostraping itself into the hidden div along with external resources
- Boostrap Done -> Switch hidden view up and replay user interactions.

Angular CLI

Guides your application throughout:

- ng new <app>
  - Scaffold best practice applications.
- ng generate
  - Generate components, services, pipes, routes, etc.
- ng serve --open
  - hot reload support
- ng build --pod
  - AOT

Future: Smaller, Faster, Easier to Use


ionic

Why build mobile apps with Angular

- Code and skill reuse.
- TypeScript
- Search engine - crawl your web app.
- Web updates usually faster.
- Progressive web gives you audience reach while native gives you richer experience.

Ionic Framework: Open source mobile SDK for developing progressive web apps and native apps.

Ionic Framework

- Progressive Web App - A mobile web app that feels like a native app.
  - Better **adoption** : no need to download.
  - Better **sharing**: a url.
- Service Worker
  - JavaScript on a separate thread.
  - Background context for an app to handle various operations.
    - Caches data for offline viewing
    - Push notifications when your web app goes away.

Ionic Framework

Angular type decorators like @Page and @App ensures UX guideline for iOS and Android respectively.

Ionic Framework

- Native device features: bluetooth, fingerprint auth, etc.
- Cordova for native deployment.
- Ready-made components. (Similar to Angular Material)
- Native Scrolling
- Hardware Accelerated Animation
- Smooth Transition
- Touch Events
- Ahead of Time Compilations (AOT) powered by Angular.


iOS

A runtime for building and running native iOS and Android apps with a single Angular/TypeScript code base.

Telerik **N**ative **S**cript

- **Virtual Machine** creates run-time to compile TS/JS to native code.
- Direct Access to native API. No wrappers (unlike ReactNative).

Native Script

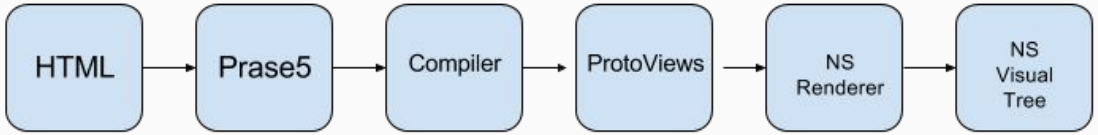
- Native UI elements. No DOM. Template XML
  - <ActionBar> - Mobile specified UI
  - <TextField>, <Label>, <StackLayout> - common UI
- Same Angular template, component, directive, input, and event syntax

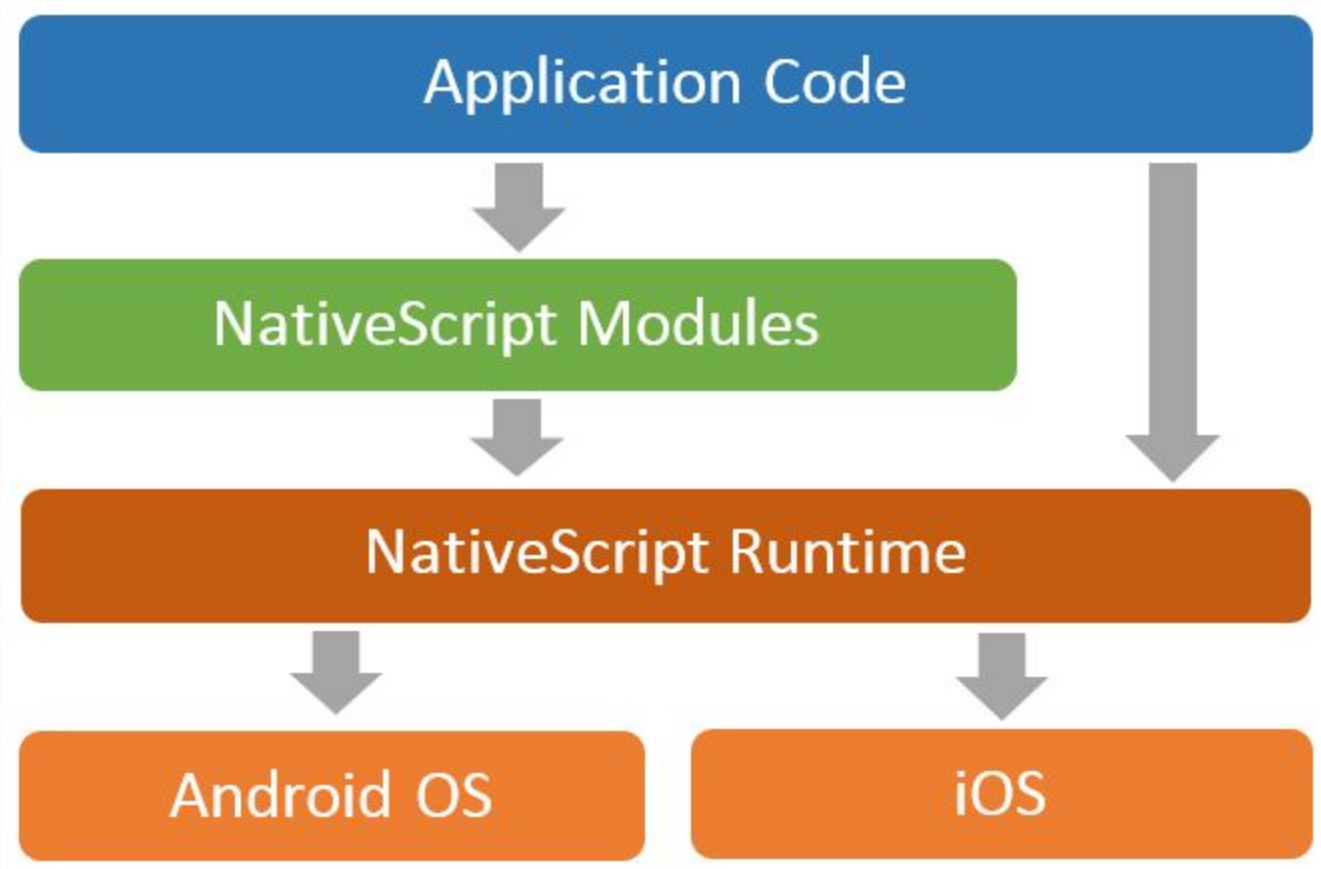
NativeScript Template

```
<StackLayout class="page">
    <Label text="{{listname}}" class="font-weight-bold text-center"></Label>
    <Image [src]="imageAsset" stretch ="none"></Image>
    <ListView [items]="items" class="list-group">
        <ng-template let-item="item">
            <Label [nsRouterLink]="['/item', item.id]" [text]="item.title"
                class="list-group-item"></Label>
        </ng-template>
    </ListView>
    <Button text="More..." (tap)="loadMore()"></Button>
</StackLayout>
```

Native Script 煊染

- Angular render compiler based on XML rather than DOM
- Angular renderer instantiate and modify NativeScript views.

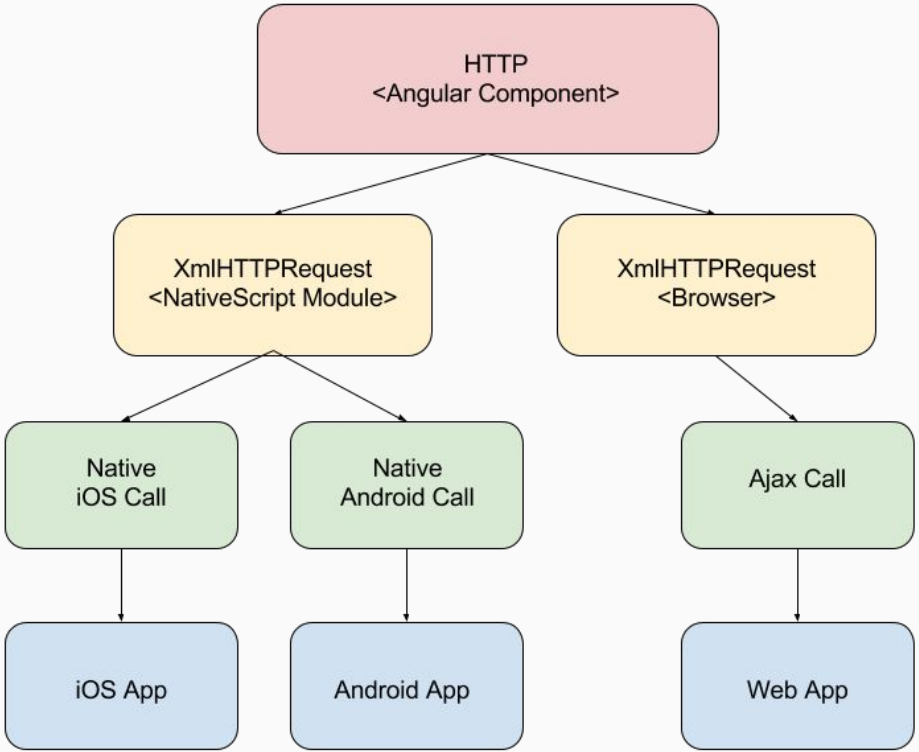



Application Code
NativeScript Modules
NativeScript Runtime
Android OS
iOS

Native Script

- **Type Conversion Service** - Converts JavaScript type to native type (Java or Objective C)
- **Metadata** - Lookup actual native platform call for the one you try to call.
- **Call Dispatcher** - Makes actual call.
- **Custom Proxy Object** - a wrapper on the native object for back and forth.

NativeScript Delegation



Native Script

- tns CLI
  - Scaffold various components
  - Live reload, build, and test.
- Plugins
  - Access native features, such as Camera
  - Npm modules, Cocoa pods, and Android Gradle, etc.
- 3rd Party JS Libraries: Integratable if no DOM dependency.
  - Typings are extractable from node_modules.
- Chrome Developer Tools

What about native code?

TNS Native Code

```
import * as Platform from "platform";
import * as Application from "application";

const pkg = Application.android.context.getPackageManager().getPackageInfo(
    Application.android.context.getPackageName(),
    PackageManager.GET_META_DATA);
java.lang.Integer.toString(pkg.versionCode);

NSBundle.mainBundle.objectForInfoDictionaryKey("CFBundleShortVersionString");
```

{{demo}}

Thanks and References

- The Google Angular Team, Stephen Fluin.
- Typescript: https://www.typescriptlang.org/
- Angular: https://angular.io
  - Victor Savkin https://vsavkin.com/
  - 中文网: https://angular.cn
  - Material: https://material.angular.io
- Ionic Framework: https://ionicframework.com
- NativeScript: https://www.nativescript.org/

End of Talk 谢谢