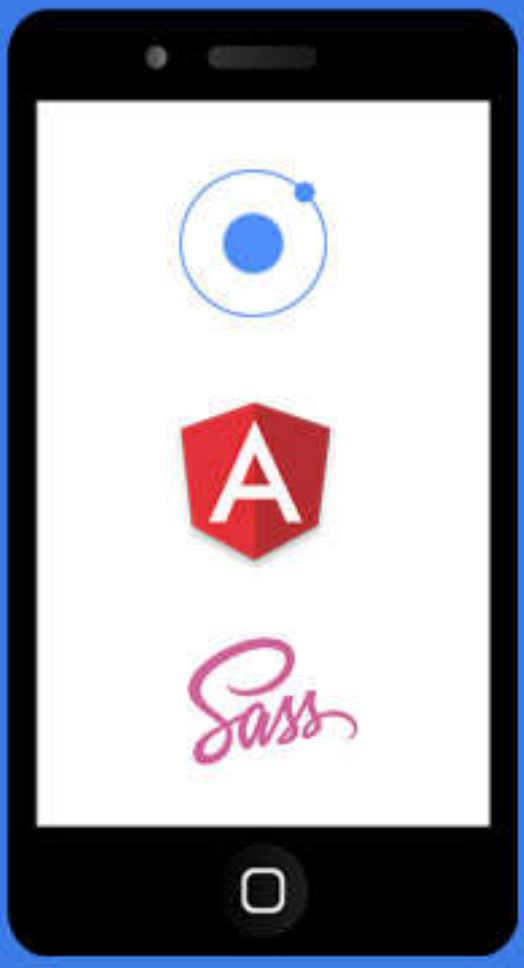
e-Book

Ionic 2

Build Mobile App for Web Developer



โค้ชเอก

Ionic 2: Build Mobile App for Web Developer

สร้าง Mobile App ด้วย Ionic Framework 2

โค้ชเอก

หนังสือเล่มนี้จำหน่ายที่ http://www.codingthailand.com/ionic2ebook

เวอร์ชัน 1 ออกจำหน่ายวันที่ 30 มิถุนายน 2559

หนังสือเล่มนี้ผู้เขียนตั้งใจจัดทำขึ้นเพื่ออยากให้มีหนังสือภาษาไทยซักเล่มเกี่ยวกับ Ionic Framework 2 ที่เน้นเนื้อหาตั้งแต่พื้นฐาน และเน้นให้ผู้อ่านได้ทำความเข้าใจถึงกระบวนการการพัฒนา Mobile App เครื่องมือต่างๆ รวมถึงภาษาโปรแกรม ไลบรารี ไม่ว่าจะเป็น Node.js, Angular 2, TypeScript และ SASS เป็นต้น นอกจากความรู้สำหรับสร้าง Mobile App แล้ว ผู้อ่านยังจะได้เรียนรู้เทคโนโลยีใหม่ๆควบคู่กันไปด้วย หวังว่าหนังสือเล่มนี้จะเป็นประโยชน์ ประหยัดเวลาการเรียนรู้ให้กับทุกคนนะครับ

“จงเอาชนะความไม่รู้ ด้วยการพัฒนาตัวเอง และลงมือทำอย่างสม่ำเสมอ”

สารบัญ

บทที่ 4 แนะนำ Ionic Framework 2

- โครงสร้างไฟล์ และโฟลเดอร์ต่างๆในโปรเจค
- การใช้ IONIC CLI COMMANDS เพื่อสร้างโค้ดอัตโนมัติ

บทที่ 5 การสร้าง Pages, Basic Navigation และ Passing Data

- การสร้าง Pages
- ทดลองสร้างเพจชื่อว่า About
- ทดลองสร้างเมนู “เกี่ยวกับเรา” เพื่อเปิดหน้าเพจ About
- พื้นฐาน Navigation
- การ Push หน้าเพจขึ้นมาแสดง
- การ Pop หน้าเพจ
- การส่งข้อมูลระหว่างเพจ (Passing Data between Pages)

บทที่ 6 Ionic 2 Components

- Components คืออะไร
- การใช้ Components เบื้องต้น

บทที่ 7 การเรียกใช้ API ด้วย Angular 2 Services

- ขั้นตอนการใช้ Angular 2 Services หรือ HTTP Services ใน Ionic 2
- การสร้าง Pages ในรูปแบบ Master Detail
- การใช้ Component Searchbar สำหรับค้นหาข้อมูล

บทที่ 8 การสร้างฟอร์ม (Form) และ Dialogs

- ลองสร้างฟอร์มสมัครสมาชิก
- การตรวจสอบความถูกต้องของข้อมูล (Validations)
- การเพิ่มข้อมูลไปที่ Backend

บทที่ 9 การใช้ SQLite สำหรับเก็บข้อมูลแบบ Local Storage

- การใช้งาน SqlStorage
- การใช้คำสั่ง SQL

บทที่ 10 การตกแต่ง App ด้วย Sass และ จัดรูปแบบการแสดงข้อมูลด้วย Pipes

- การทำ Theming ให้กับ App
- Utility Attributes
- การตกแต่ง App โดยการเขียนทับตัวแปร Sass ของ Ionic
- จัดรูปแบบการแสดงข้อมูลด้วย Pipes
- พื้นฐานการใช้งาน Built-in pipes ใน Angular 2
- การสร้าง Custom Pipes ใน Ionic 2

บทที่ 11 การเตรียม Resources และ Publish App

- การเปิด Production Mode
- การออกแบบไอคอน และ splash screen ให้กับ App
- ตั้งค่า Bundle ID และ App Name สำหรับ App ที่ Build ด้วย Ionic
- การลดขนาดภาพ
- การ Sign applications
- การ Sign Android Applications
- การ Build iOS และ Android App ด้วย PHONEGAP BUILD
- การ Build Android App ด้วย Ionic CLI
- การนำไฟล์ที่ Build เสร็จแล้วอัพโหลดขึ้น Store

บทที่ 12 โบนัสพิเศษ

- ตัวอย่างไฟล์การตั้งค่า App เพื่ออัพโหลดไปยัง PhoneGap Build
- ตัวอย่างไฟล์ psd (PhotoShop) สำหรับทำ splash screen
- ไฟล์โค้ดประกอบหนังสือ

Changelog

เวอร์ชัน 2 – 30 มิถุนายน 2559

- Ionic Framework 2 beta.10
- Angular 2 rc.3

เวอร์ชัน 1 – 16 มิถุนายน 2559

- Ionic Framework 2 beta.9
- Angular 2 rc.1

Ionic Framework 2 เป็น Framework สมัยใหม่ที่ช่วยเราพัฒนา Mobile App ได้อย่างรวดเร็ว และง่ายมากๆ แนวคิดคือ การพัฒนา Mobile App ให้เหมือนกับพัฒนา Web App นั่นเอง โดยที่นักพัฒนาแต่ละคนจะต้องมีความรู้พื้นฐาน HTML, CSS และ JavaScript (JS ES6) มาแล้วถึงจะสามารถเรียนรู้ Ionic ได้อย่างรวดเร็ว

Ionic Framework ในเวอร์ชัน 2 นี้ ข้อดีของผู้เริ่มต้นคือ ไม่จำเป็นต้องรู้ Ionic เวอร์ชัน 1 มาก่อนก็สามารถเรียนรู้ได้ทันที เพราะตัว Ionic 2 เอง ได้ถูกเขียนโค้ดขึ้นมาใหม่ทั้งหมดด้วย Angular 2 และ TypeScript ครับ เพราะฉะนั้นสิ่งที่เป็นพื้นฐานที่ขาดไม่ได้เลยหากผู้เรียนต้องการสร้าง Mobile App ที่ซับซ้อนมากขึ้น ก็ขอแนะนำว่าให้ศึกษา Angular 2 ด้วยนะ มาลุยกันเลยครับ!

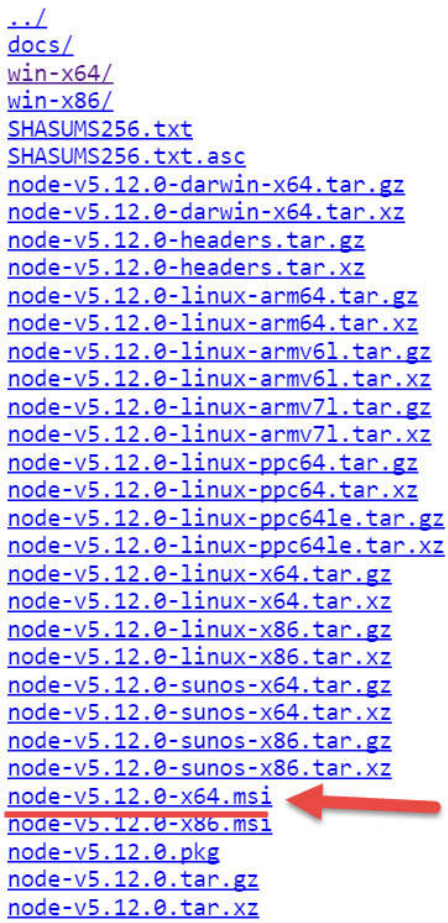
บทที่ 1 การติดตั้ง Node.js, npm and Atom Editor

ก่อนเขียน Ionic 2 เราต้องติดตั้งซอฟต์แวร์ และเครื่องมือกันก่อน ประกอบด้วย Node.js, npm และ Atom Editor

การติดตั้ง Node.js และ mpm

การติดตั้ง Node.js สามารถดาวน์โหลดการติดตั้งได้โดยตรงที่เว็บ https://nodejs.org/en/download/current/ เราสามารถติดตั้งได้ทั้ง Windows, Linux และ Mac ในหนังสือเล่มนี้จะใช้ Node.js สำหรับ Windows เป็นหลัก แต่หากใครใช้ Mac หรือ Linux ก็ใช้ได้ไม่มีปัญหา เพราะคำสั่งต่างๆที่ใช้กับ Ionic เหมือนกันทุกอย่าง

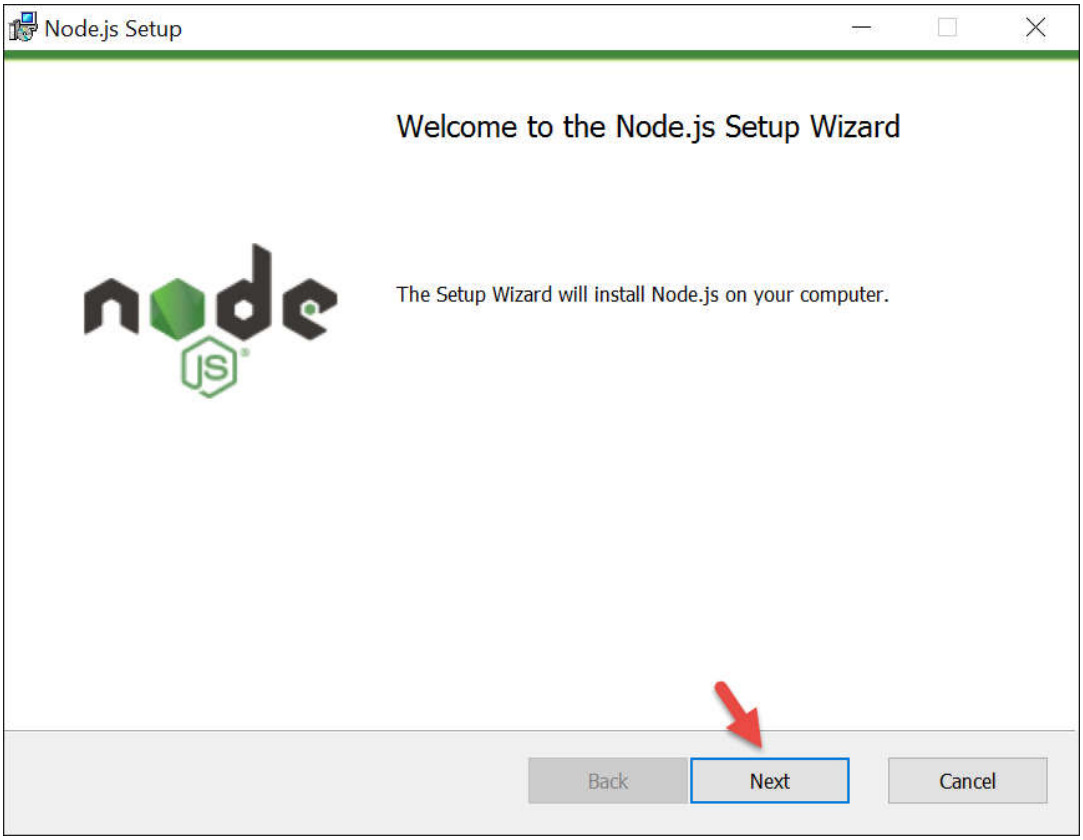
ณ ขณะที่เขียนหนังสืออยู่นี้ Ionic Framework 2 แนะนำให้เราใช้ *Node.js เวอร์ชัน 5.x และ npm เวอร์ชัน 3.x* ครับ ดาวน์โหลดได้ที่ https://nodejs.org/dist/latest-v5.x/ สำหรับคนที่ใช้ Windows 64 bit ให้เลือก x64 ส่วน Windows 32 bit ก็ให้เลือก x86



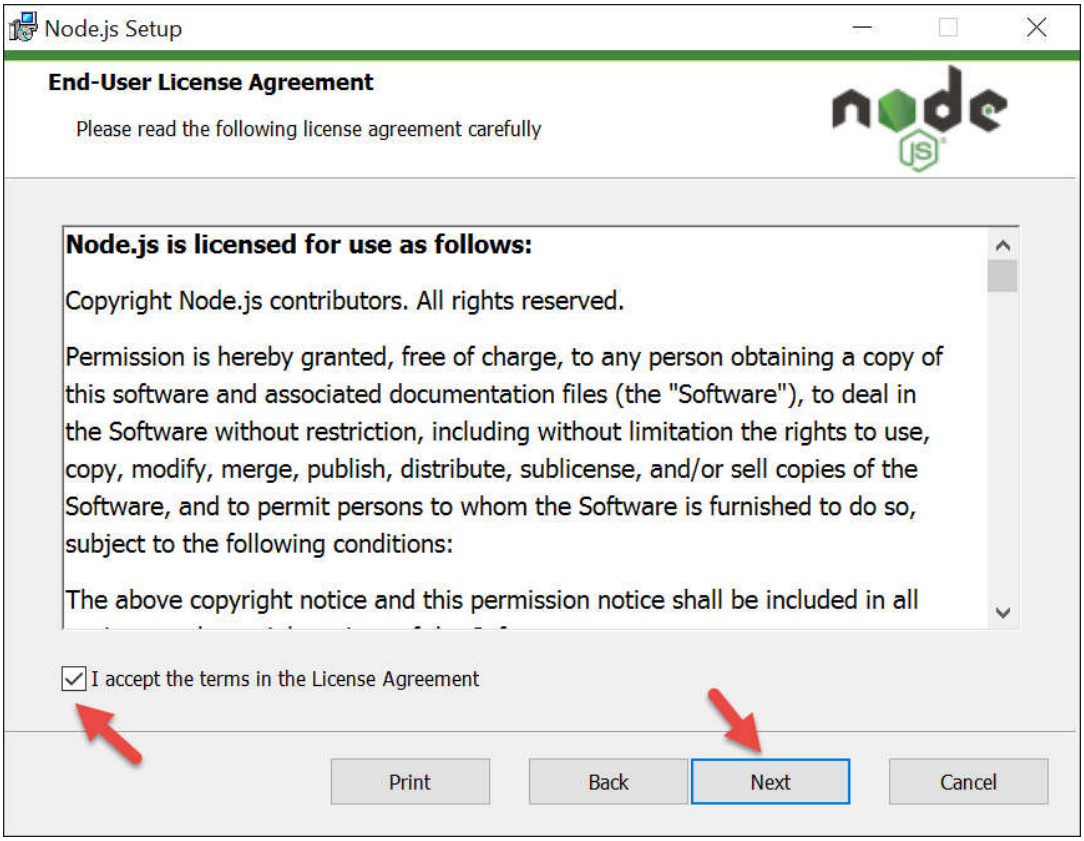
Note: จริงๆแล้วสามารถติดตั้ง Node.js เวอร์ชันล่าสุดได้ แต่บางครั้งจะมีปัญหา Library บางตัวรองรับไม่ทัน แนะนำให้อย่าเพิ่งเลือกเวอร์ชันล่าสุดครับ ให้ถอยเวอร์ชันลงมา 1 เวอร์ชัน เช่น ขณะที่เขียนหนังสืออยู่นี้เป็นเวอร์ชัน 6 แนะนำให้ใช้เวอร์ชัน 5 ไปก่อน

หลังจากดาวน์โหลดมาแล้วให้ติดตั้งตามขั้นตอน ดังนี้

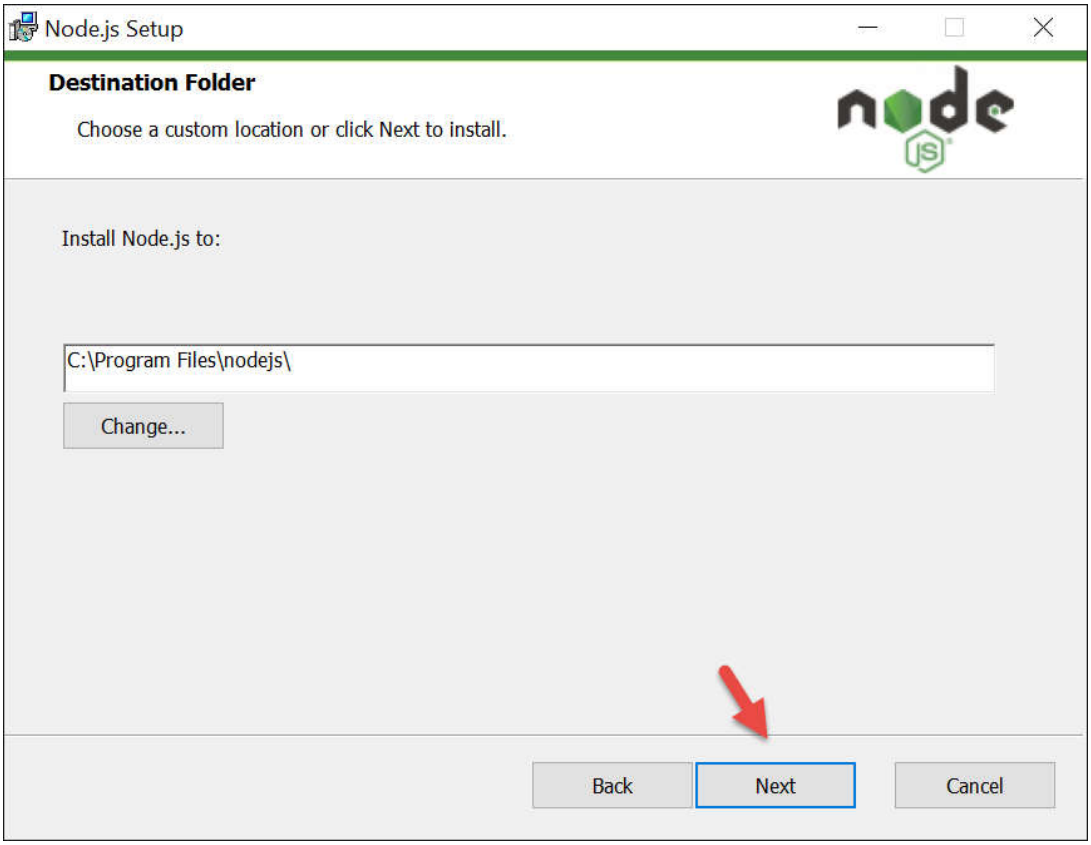
1.ดับเบิ้ลคลิกไฟล์ติดตั้ง แล้วกดปุ่ม Next



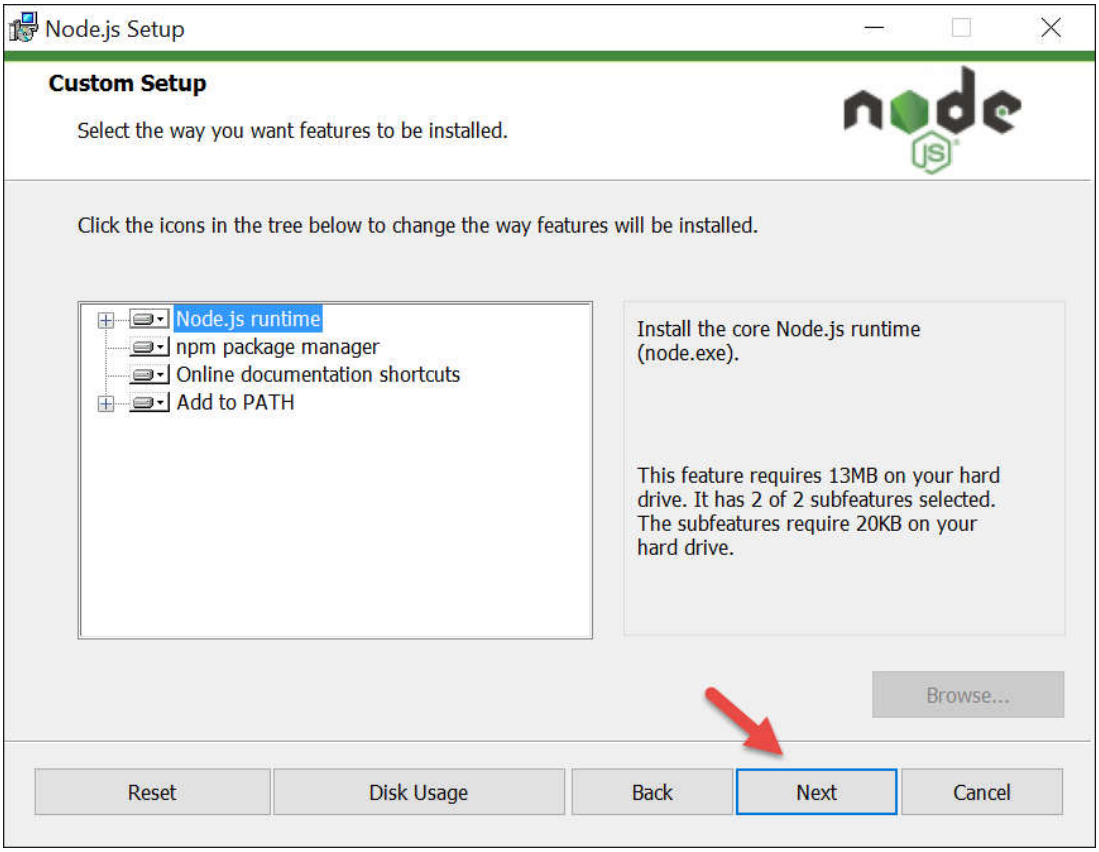
2.คลิกเลือก I accept the terms… แล้วกดปุ่ม Next



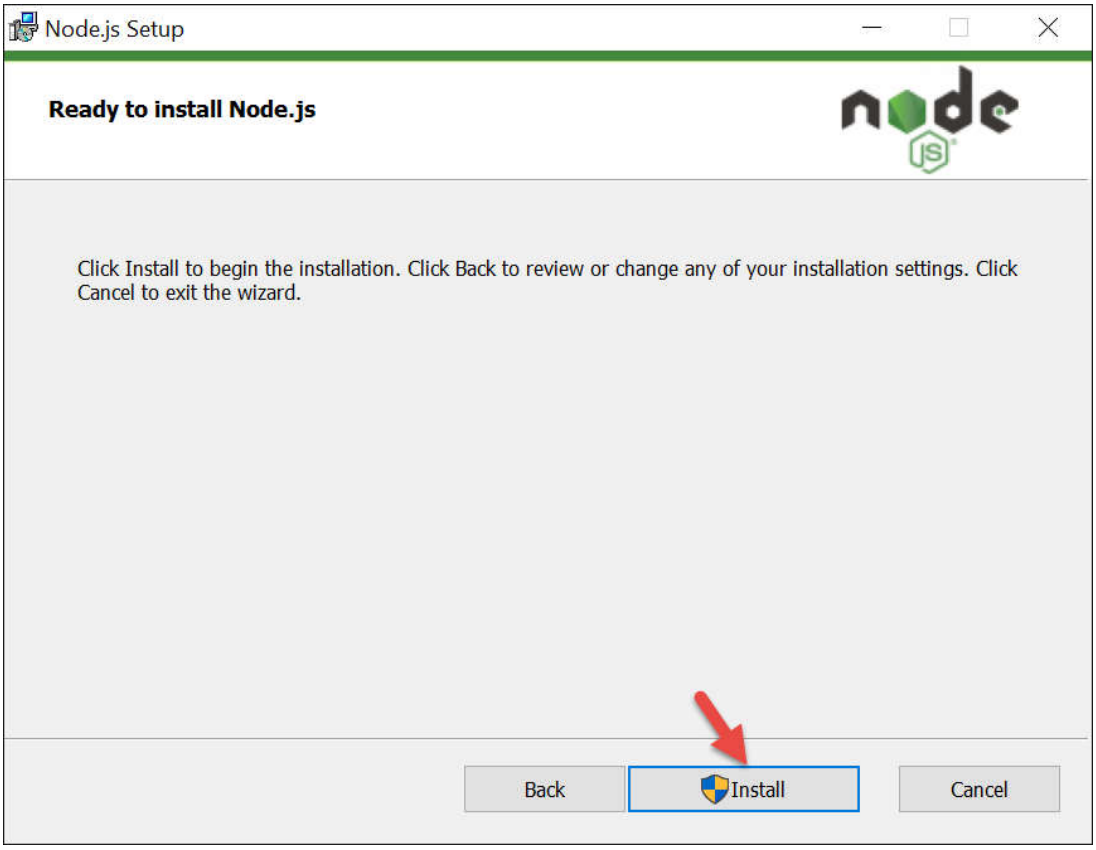
3.กดปุ่ม Next



4. กดปุ่ม Next (แนะนำว่าไม่ต้องแก้อะไร)

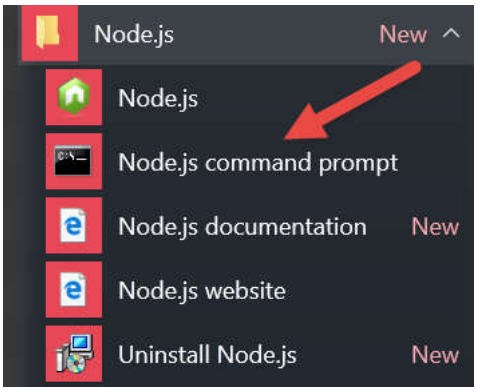


5.คลิกปุ่ม Install เพื่อติดตั้ง Node.js

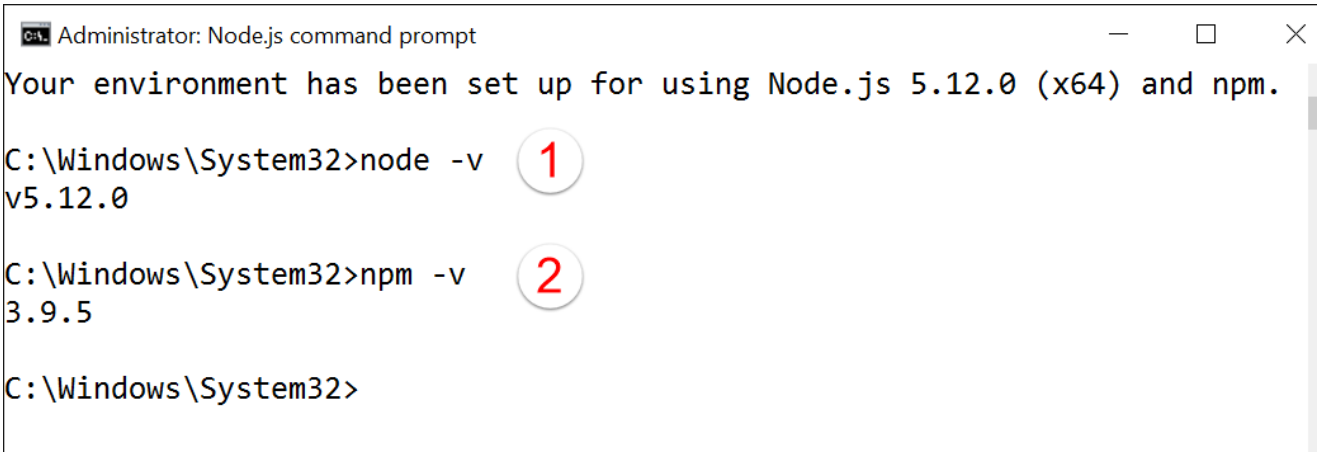


การตรวจสอบเวอร์ชัน Node.js และ npm

หลังจากติดตั้ง Node.js เรียบร้อยแล้วให้ตรวจสอบเวอร์ชันของ Node.js และ npm ด้วย และเป็นอีกทางเพื่อตรวจสอบว่าเราได้ติดตั้ง Node.js สมบูรณ์พร้อมใช้งานได้หรือไม่ ให้เปิด **Node.js command prompt** (จะค้นหา หรือคลิกที่ Start Menu ของ Windows ก็ได้)



เมื่อเปิดขึ้นมาแล้วให้พิมพ์คำสั่ง **node -v** แล้วกด enter เพื่อตรวจสอบเวอร์ชันของ Node.js และพิมพ์ **npm -v** แล้วกด Enter เพื่อตรวจสอบเวอร์ชันของ npm



Note: การเปิดใช้งาน Node.js command prompt ทุกครั้ง แนะนำให้คลิกขวา แล้วเลือก Run as administrator ทุกครั้ง

รู้จักกับ npm

npm เป็นตัวจัดการ dependencies หรือ library ต่างๆ ในโปรเจคของเรา ข้อดีคือ เราไม่ต้องเข้าเว็บเพื่อไปดาวน์โหลดไฟล์แล้วติดตั้งด้วยตัวเอง เราสามารถใช้ npm จัดการติดตั้ง หรือลบ dependencies/library ต่างๆได้อย่างสะดวก (ผ่านอินเทอร์เน็ต)

สรุปคำสั่ง npm พื้นฐาน

- npm init เริ่มต้นใช้งาน และช่วยให้เราสร้าง และใส่รายละเอียดเกี่ยวกับโปรเจคของเรา (package.json)
- npm install module_name ติดตั้ง module
- npm install -g module_name ติดตั้ง module ในระดับ global
- npm install module_name --save ติดตั้ง module และเพิ่มมันเข้าไปในไฟล์ package.json (ข้างใน dependencies)
- npm install module_name --save-dev เหมือนกันกับข้างบน ติดตั้ง module และเพิ่มมันเข้าไปในไฟล์ package.json (ข้างใน dependencies) และบอกว่าจะนำมาช่วยในส่วนของนักพัฒนา
- npm list แสดงรายการ modules ทั้งหมดที่ติดตั้งไปแล้วในโปรเจคของเรา
- npm list -g แสดงรายการ modules ระดับ global ทั้งหมดที่ถูกติดตั้งบน OS ของเรา
- npm remove module_name uninstall module ออกจากโปรเจคของเรา
- npm remove -g module_name uninstall module ระดับ global
- npm remove module_name --save uninstall module ออกจากโปรเจคของเรา พร้อมทั้งนำออกจาก attribute dependencies ในไฟล์ package.json
- npm remove module_name --save-dev เหมือนกันกับด้านบน uninstall module ออกจากโปรเจคของเรา พร้อมทั้งนำออกจาก attribute dependencies ในไฟล์ package.json
- npm update module_name update เวอร์ชัน module ให้ใหม่ล่าสุด

- npm update -g module_name update เวอร์ชัน module ให้ใหม่ล่าสุด ในระดับ global
- npm -v แสดงเวอร์ชันปัจจุบันของ npm

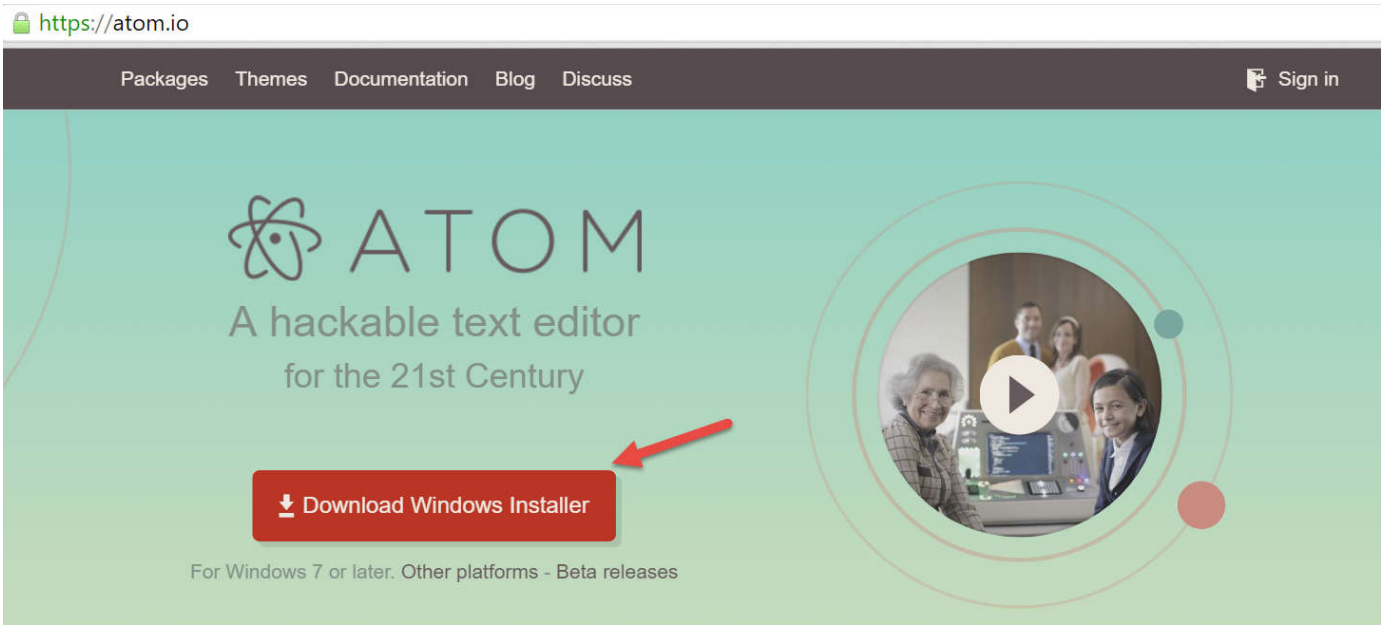
> **Note:** เว็บไซต์หลักของ mpm https://www.npmjs.com/

การติดตั้ง Atom Editor

Atom เป็น text editor สมัยใหม่ โดยมีคุณสมบัติเด่น คือ Cross-platform และมีตัวจัดการ Packages หรือส่วนเสริมเพื่อช่วยให้นักพัฒนาเขียนโค้ดได้ง่ายขึ้น

> **Note:** เว็บไซต์หลัก Atom https://atom.io เราสามารถหา Packages ต่างๆเพื่อช่วยให้เขียนโค้ดง่ายขึ้นได้ที่ https://atom.io/packages

เนื่องจากการติดตั้ง Atom Editor ค่อนข้างง่ายเพียงแค่กดปุ่ม Next ก็เสร็จเรียบร้อยจึงไม่ขอกล่าวถึงในหนังสือเล่มนี้แต่ถ้าหากสงสัยค่อยถามมาทีหลังได้ครับ



การติดตั้ง Packages ให้กับ Atom Editor

สำหรับการเขียน Ionic มี Packages ที่เราควรติดตั้ง มีดังนี้

atom-typescript ใช้สำหรับช่วยเขียน TypeScript สามารถตรวจสอบข้อผิดพลาดต่างๆให้เราได้ (อันนี้ต้องติดตั้ง)

https://atom.io/packages/atom-typescript

minimap ใช้สำหรับช่วยดูตัวอย่างของโค้ดทั้งไฟล์

https://atom.io/packages/minimap

file-icons ใช้สำหรับแสดงไอคอนไฟล์ต่างๆในโปรเจคให้ดูง่าย และสวยงาม

https://atom.io/packages/file-icons

atom-beautify ใช้สำหรับจัดรูปแบบโค้ดให้มีรูปแบบที่สวยงาม

https://atom.io/packages/atom-beautify

linter ใช้สำหรับช่วยแสดงข้อความ errors ต่างๆที่เกิดขึ้น

https://atom.io/packages/linter

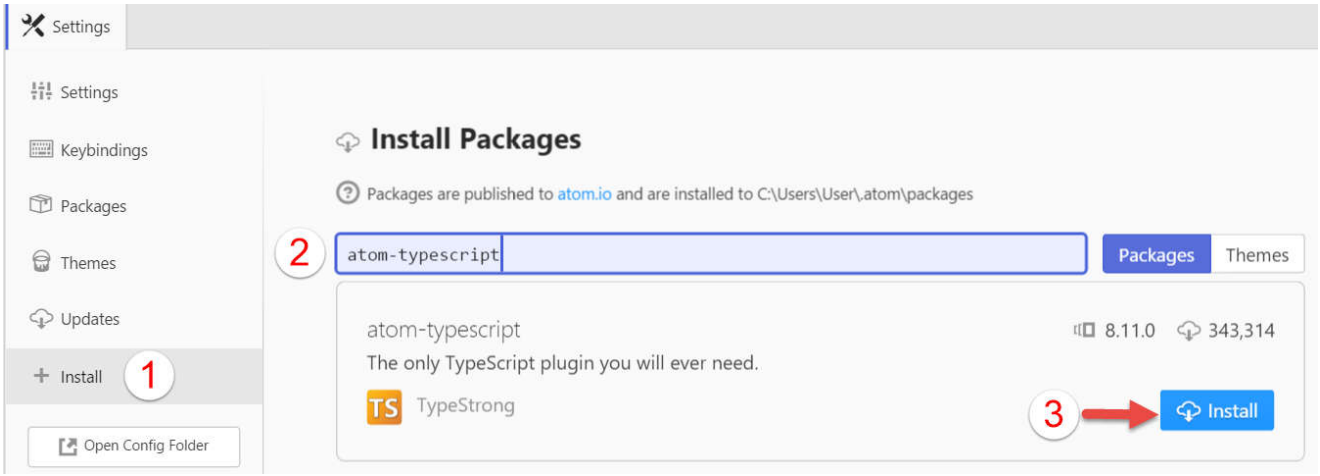
color-picker ใช้สำหรับเลือกสีเพื่อตกแต่งหน้าเว็บได้สะดวกขึ้น

https://atom.io/packages/color-picker

ขั้นตอนการติดตั้ง Packages ต่างๆ มีดังนี้ (ยกตัวอย่าง atom-typescript ส่วนตัวอื่นๆใช้วิธีเดียวกันครับ)

1.เปิด Atom Editor ขึ้นมา คลิกเลือกเมนู File -> Settings

2.คลิกที่ Install พิมพ์ชื่อ Package ที่ต้องการติดตั้งในช่อง Search packages เช่น **atom-typescript** แล้วกด **Enter**



3.กดปุ่ม Install แค่นี้ก็เรียบร้อย จากนั้นให้ติดตั้ง Packages ให้ครบทุกตัวที่แนะนำไปด้านบนนะครับ

Note: หากกดปุ่ม Install ไปแล้วรอนานหรือมีข้อผิดพลาด แนะนำให้ปิด Atom แล้วเปิดใหม่อีกครั้ง

Note: หากใช้ Atom Editor แล้วมีปัญหา สามารถใช้ Visual Studio Code แทนได้ (https://code.visualstudio.com/)

บทที่ 2 พื้นฐาน Angular 2, TypeScript และ Sass สำหรับ Ionic 2

พื้นฐาน Angular 2

Angular 2 เป็น JavaScript framework หรือเรียกได้ว่าเป็น platform สำหรับสร้าง applications ฝั่ง client และเขียนด้วย TypeScript ในการพัฒนา Ionic 2 นั้น เราจะต้องเขียนโค้ดด้วย Angular 2 (TypeScript) ดังนั้นจึงจำเป็นอย่างมากที่เราจะต้องเรียนรู้ Angular 2 ด้วยครับ

Angular 2 Modules

Module เป็นแนวคิดในการเขียนโค้ดชุดหนึ่งขึ้นมาใช้งานด้วยจุดประสงค์บางอย่าง เราสามารถ import module เข้ามาใช้งานได้ตลอดเวลา จะเป็น Module ที่เราเขียนขึ้นมาเอง หรือจะ import library modules ของ Angular 2 เข้ามาใช้ก็ได้ ยกตัวอย่าง เช่น

```
import {Component} from '@angular/core';
import {MyDetailsPage} from '../my-details/my-details';

@Component({
 templateUrl: 'build/pages/home/home.html'
})

export class HomePage {
constructor(){ }
}
```

จากตัวอย่างโค้ดเป็นนำเข้า (import) Angular Component function จาก module @angular/core และหากเราต้องการส่งออก (export) ฟังก์ชันหรือคลาส ก็ให้เขียน export ไว้ด้านหน้า

สรุปเรื่อง Modules

1. App ที่เขียนด้วย Angular 2 ถูกประกอบขึ้นมาด้วย Modules ต่างๆ ซึ่งทำหน้าที่แตกต่างกัน
2. Modules จะ export บางอย่าง ได้แก่ classes, function, values แต่ละ modules อาจมีการ import modules ต่างๆเข้ามาใช้งานได้
3. การเขียน Application ด้วย Angular 2 ก็คือ การเขียนเพื่อเรียกใช้ Modules ต่างๆ นั่นเอง

Note: หลักการ import และ export นี้ เป็นหลักการของ JavaScript (ES2015) หรือ ES6

Angular 2 Decorators

Decorators เป็นฟังก์ชันที่เราสามารถเพิ่ม meta data ต่างๆให้กับ Class อาจเป็นสมาชิกของคลาสหรือ method ของคลาสก็ได้ Decorators จะขึ้นต้นด้วยเครื่องหมาย @ เช่น @Component, @Pipe เป็นต้น

Angular 2 Components

ใน Angular 2 นั้น Component คือ ส่วนที่ใช้ในการควบคุมการแสดงผล หรือเรียกว่าง่ายๆว่า ส่วน view ประกอบด้วย Template, Class (properties + methods) และ meta data ที่เกี่ยวข้องส่วนของการแสดงผล และ logic ต่างๆ ที่ใช้ในหน้าเพจ ส่วนประกอบ Components มีดังนี้

Note: ใน Ionic 2 หากเปรียบเทียบง่ายๆ 1 Component ก็คือ 1 หน้า ของ Mobile App นั่นเอง

พื้นฐาน TypeScript

TypeScript เป็นส่วนขยาย และเป็น superset ของภาษา JavaScript โดยเราสามารถเขียนระบุชนิดข้อมูล หรือ type ได้ สามารถเขียน Classes, Interfaces หรือ Generics ได้ (คล้ายการเขียน c# หรือ java) แน่นอน Ionic 2 ใช้ TypeScript ในการเขียนโค้ดเป็นหลักครับ

Note: หากอยากศึกษา TypeScript เพิ่มเติม แนะนำเว็บ https://www.typescriptlang.org/docs/handbook/basic-types.html

ชนิดข้อมูลของ TypeScript และการประกาศตัวแปร

ชนิดข้อมูลของ TypeScript คล้ายกับ JavaScript ประกอบชนิดข้อมูลที่ใช้บ่อย ดังนี้

1. **Boolean** เป็นข้อมูลชนิดเก็บค่า true หรือ false

```
let isDone: boolean = false;
```

2. **Number** เป็นชนิดข้อมูลเก็บค่าตัวเลข โดยปกติแล้ว Number จะเป็น floating point

```
let decimal: number = 6;

let hex: number = 0xf00d;

let binary: number = 0b1010;

let octal: number = 0o744;
```

3. **String** เป็นชนิดข้อมูลเก็บข้อความต่างๆ โดยจะอยู่ในเครื่องหมาย quotes (") หรือ single quotes (')

```
let color: string = "red";

let fullName: string = `Bob Bobbington`;

let age: number = 37;

let sentence: string = `Hello, my name is ${ fullName }.

I'll be ${ age + 1 } years old next month.`
```

เราสามารถใช้ **template strings** ได้โดยใช้เครื่องหมาย backtick (`) ครอบ string ที่ต้องการหากต้องการแสดงตัวแปรให้ใช้ ${ }

4. **Array** เป็นชนิดข้อมูลเป็นชุดสามารถเก็บค่าได้หลายค่าในตัวแปรเดียว

```
let list: number[] = [1, 2, 3];

let list: Array<number>;

let items: Array<{}>;
```

5. **Any** เป็นชนิดข้อมูลที่เราประกาศไว้หากยังไม่รู้ว่าจะเก็บค่าอะไร หรือเรียกว่าการเก็บข้อมูลแบบ dynamic ก็ได้

```
let notSure: any = 4;

let notSure: any = null;
```

พื้นฐาน Sass (Syntactically Awesome StyleSheets)

Sass เป็น stylesheet language (จะถูก complies เป็น CSS อีกที) ใช้สำหรับตกแต่งหน้าเพจให้สวยงาม โดยมีความสามารถ เช่น variables, nesting และ mixins เป็นต้น ใน Ionic 2 เราสามารถเขียน Sass หรือจะเขียน CSS ปกติก็ได้เช่นเดียวกัน

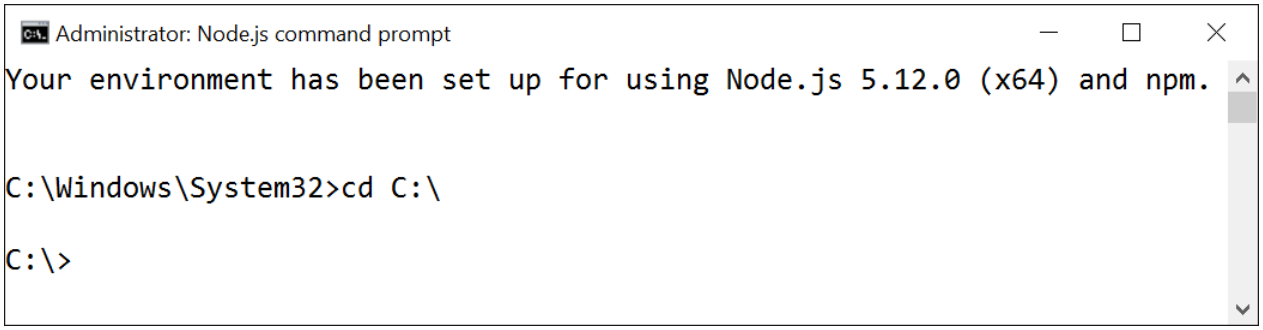
Note: หากต้องการศึกษาข้อมูลเพิ่มเติมดูได้ที่เว็บ http://sass-lang.com/

บทที่ 3 การติดตั้ง Ionic 2

การติดตั้ง Ionic Framework 2

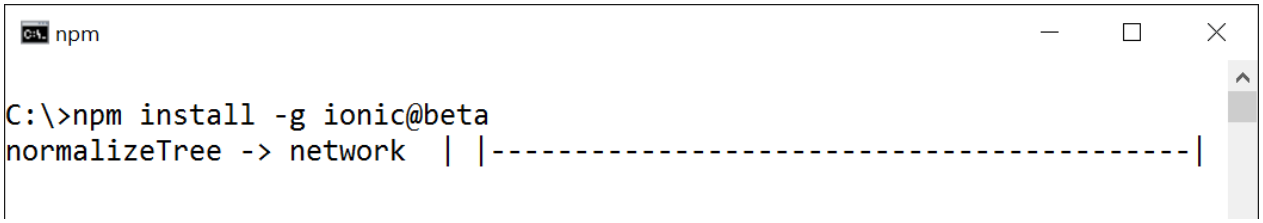
การติดตั้ง Ionic Framework 2 มีขั้นตอน ดังนี้

1. เปิด **Node.js command prompt** แล้วพิมพ์คำสั่ง **cd C:** แล้วกด Enter (จะสร้างโปรเจคใหม่ที่ Drive C สามารถเปลี่ยนได้ตามความต้องการ)



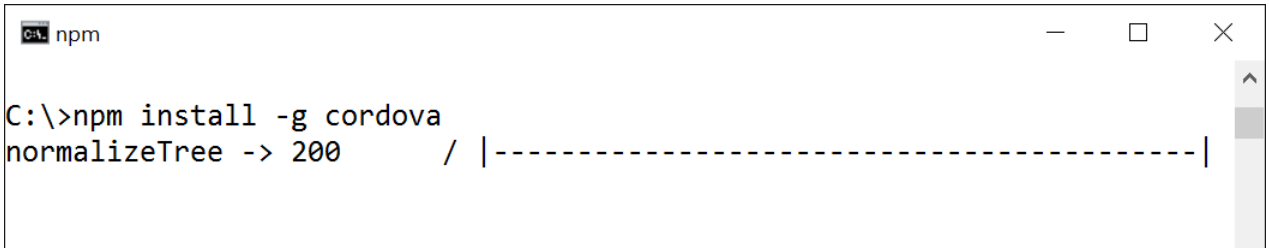
Note: ก่อนเปิด Node.js command prompt *อย่าลืม Run as administrator*

2. ติดตั้ง Ionic 2 พิมพ์คำสั่ง **npm install -g ionic@beta** แล้วกด Enter รอจนเสร็จเรียบร้อย



Note: หากใช้ MAC หรือ Linux ใช้คำสั่ง **sudo** npm install -g ionic@beta ตรง @beta หาก Ionic 2 ออกตัวจริงอาจมีการเปลี่ยนแปลงได้

3. ติดตั้ง Cordova สำหรับใช้ build ไปที่ emulator หรือเครื่องทดสอบ (USB) ใช้คำสั่ง **npm install -g cordova** แล้วกด Enter



4. ติดตั้ง Android SDK สำหรับใช้ build Android วิธีติดตั้งที่ง่ายที่สุดแนะนำให้ติดตั้ง Android Studio ซึ่งได้รวม Android SDK ไว้แล้วได้ที่ https://developer.android.com/studio/index.html

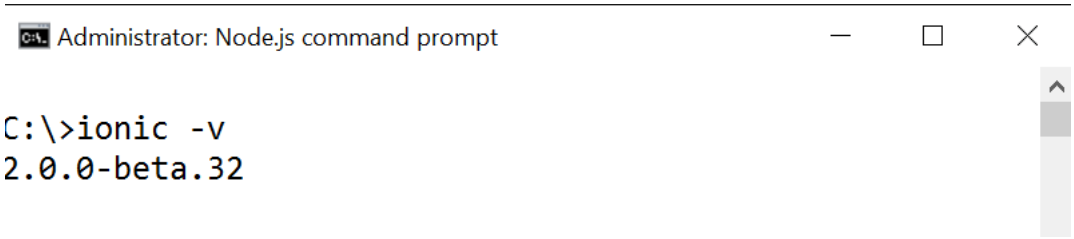
   Note: สำหรับคนที่ใช้ MAC ให้ติดตั้ง XCode เพื่อใช้ build App ตามนี้ https://developer.apple.com/xcode/

   Note: หากต้องการดูวิดีโอการติดตั้ง Ionic 2 สามารถดูได้ที่ https://goo.gl/GNmz0c

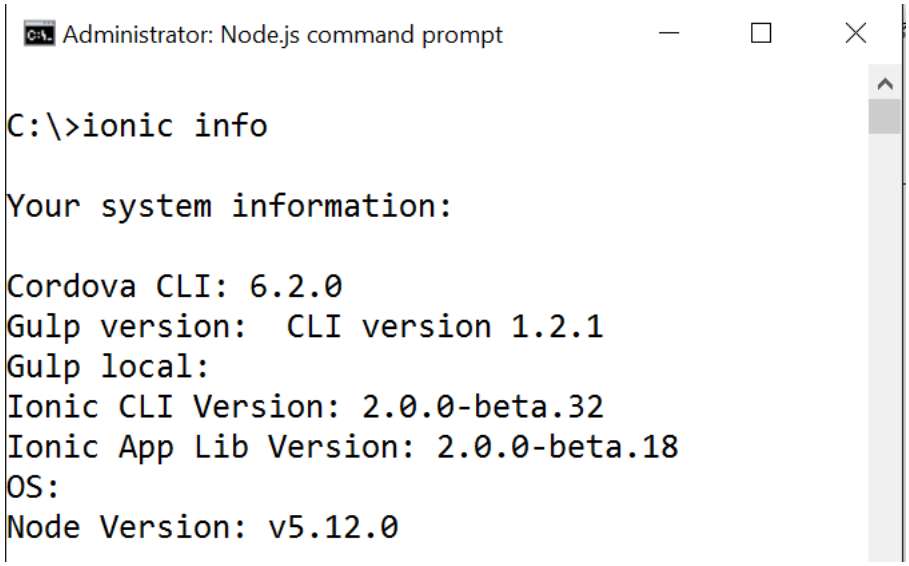
   Note: หากต้องการดูวิดีโอการติดตั้ง Android Studio ดูได้ที่ https://goo.gl/jHor3L

หากไม่ติดตั้งข้อ 4 เราสามารถทดสอบ App ผ่าน Browser เช่น Google Chrome ได้ครับ ข้อ 4 นี้สามารถติดตั้งทีหลังได้เมื่อต้องการต้องการ build App

หลังจากติดตั้ง Ionic 2 เรียบร้อย ให้ตรวจสอบเวอร์ชันของIonic CLI (Command Line Interface) ด้วยคำสั่ง **ionic -v**



หากต้องการดู Environments ต่างๆ ให้ลองใช้คำสั่ง **ionic info** แล้วกด Enter



การสร้างโปรเจคใหม่

การสร้างโปรเจคใหม่เราสามารถใช้ Ionic CLI พิมพ์คำสั่งได้เลย โดยมีรูปแบบคำสั่ง ดังนี้

ionic start <**ชื่อโปรเจค**> <template> --v2 –ts

สำหรับการสร้างโปรเจคใหม่นั้นเราสามารถใส่ templates หรือไม่ก็ได้ templates เป็นรูปแบบ App สำเร็จรูปทำให้ประหยัดเวลาในการพัฒนา ใน Ionic 2 นั้นมี Templates ให้เลือก ดังนี้

1. หากไม่ใส่ template จะเป็น App ในรูปแบบของ tabs application ใช้คำสั่ง

   ionic start MyIonic2 --v2 –ts
2. Template แบบ blank ใช้คำสั่ง

   ionic start MyIonic2 blank --v2 --ts
3. Template แบบ sidemenu ใช้คำสั่ง

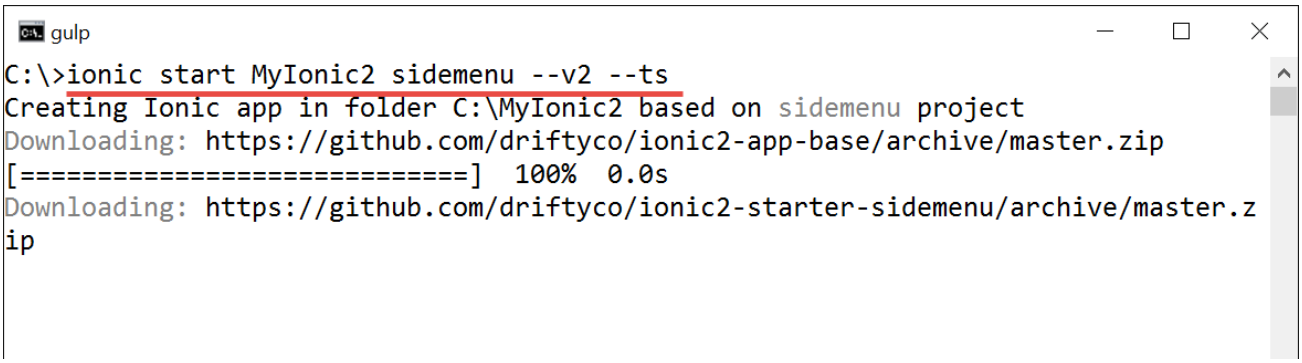
   ionic start MyIonic2 sidemenu --v2 --ts

   Note: --ts บอกว่าเราจะเขียน TypeScript

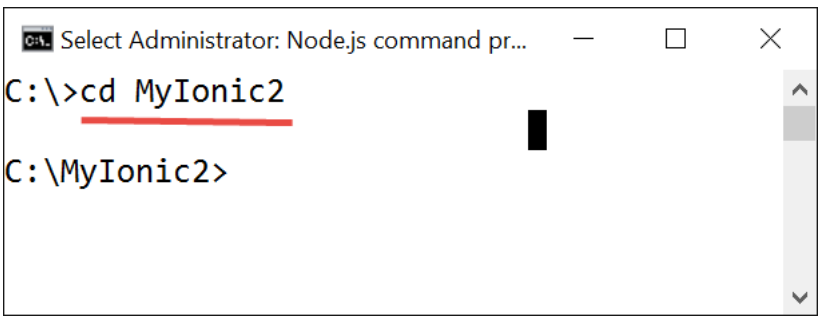
ลงมือสร้างโปรเจคใหม่

ในหนังสือเล่มนี้เราจะใช้ **Template sidemenu** และตั้งชื่อโปรเจคว่า **MyIonic2** ให้รันคำสั่งนี้ แล้วกด Enter

ionic start MyIonic2 sidemenu --v2 --ts



หลังจากสร้างโปรเจคเรียบร้อยให้ใช้คำสั่ง **cd MyIonic2** แล้วกด Enter เข้าไปยังโฟลเดอร์โปรเจคของเรา



เพิ่ม Platforms ที่เราต้องการ Build

ต่อมาเราควรเพิ่ม Platforms สำหรับ build app ถ้า Android ใช้คำสั่ง

ionic platform add android

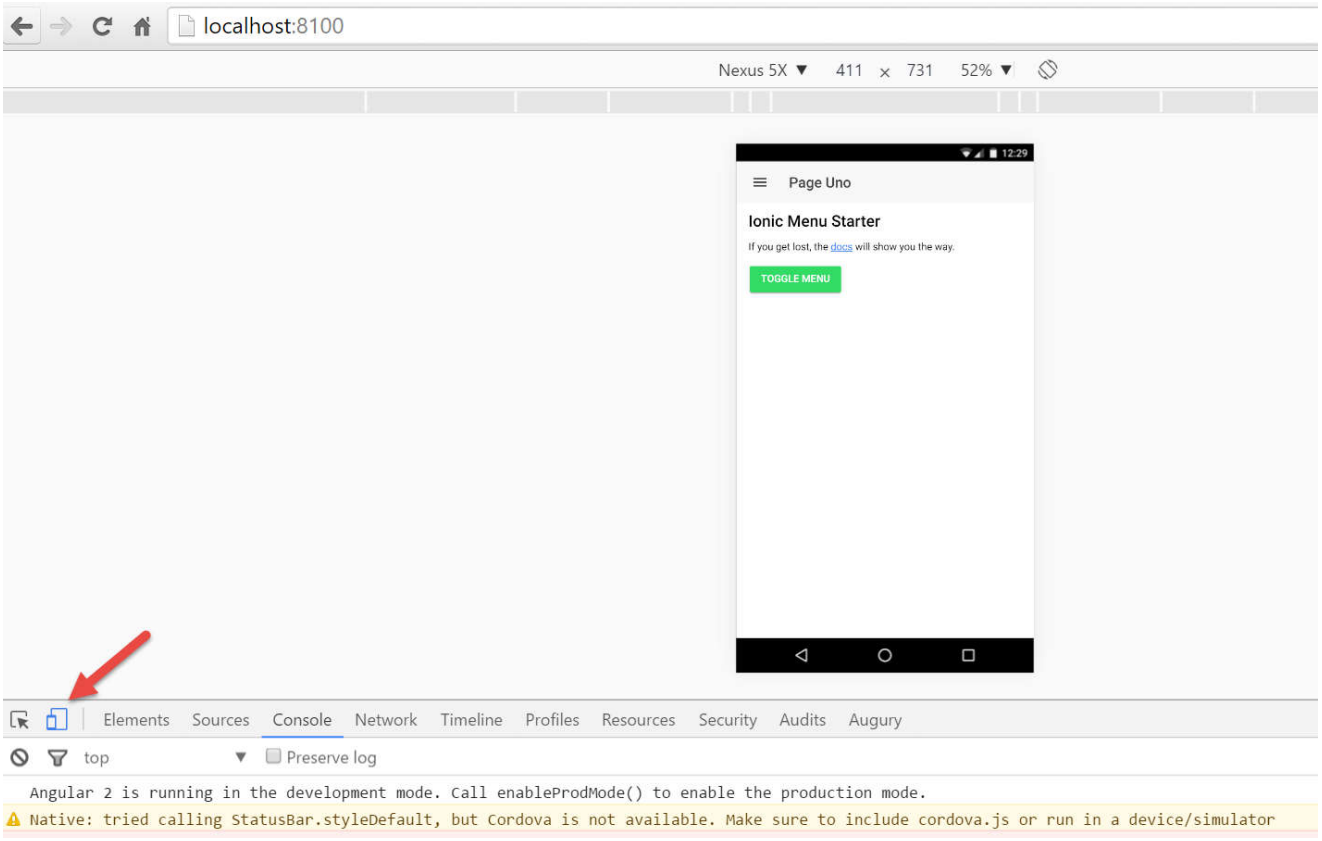
ถ้า iOS ใช้คำสั่ง

ionic platform add ios

> **Note:** ถ้าต้องการยกเลิกหรือลบ platform ให้ใช้คำสั่ง ionic platform rm android หรือ ionic platform rm ios

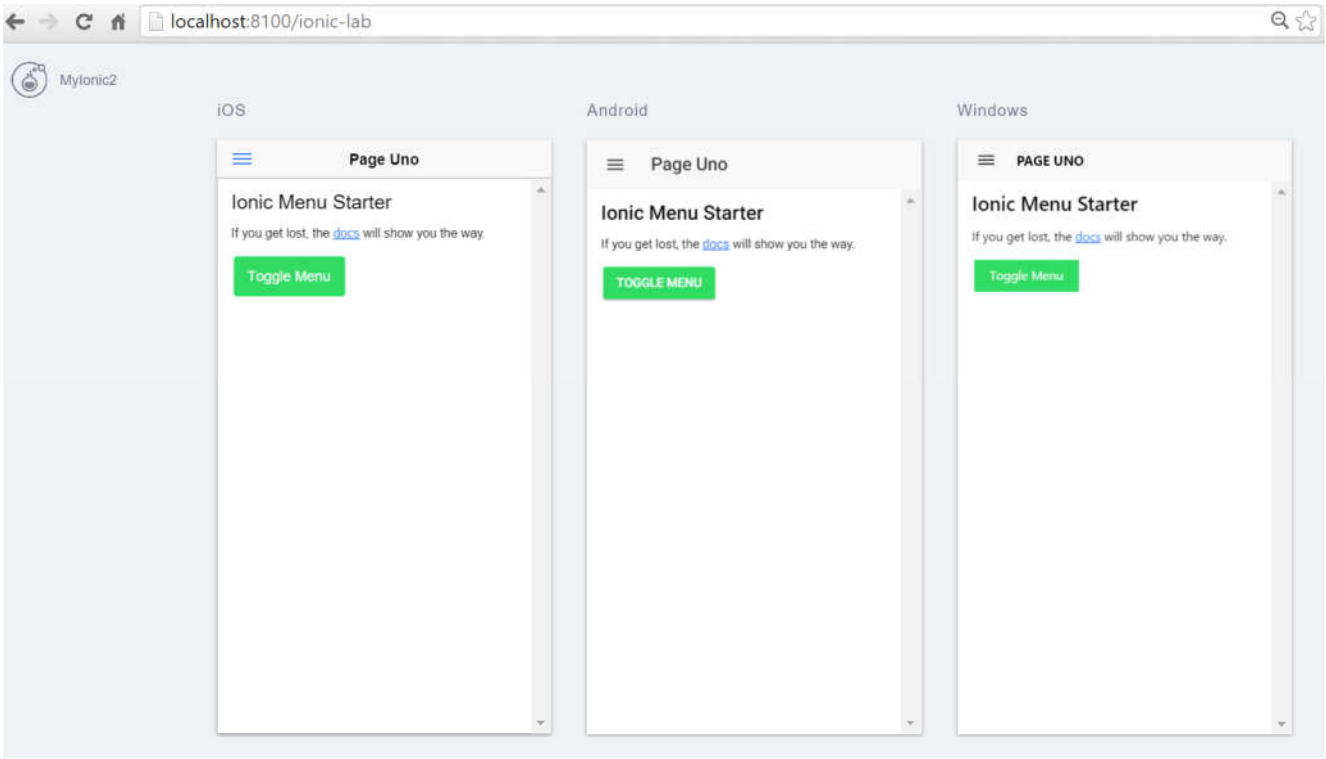
การสั่งรัน Application ไปที่ Browser

การสั่งรัน Application ให้ใช้คำสั่ง **ionic serve** แล้วกด Enter หลังจากรันคำสั่งจะมีการเปิด Web browser ขึ้นมา พร้อมกับมี Web Server มาให้ด้วย หากเราแก้ไขโค้ดแล้วบันทึกไฟล์ Web browser จะ refresh อัตโนมัติครับ ในหนังสือเล่มนี้เราจะใช้ Google Chrome เป็นหลัก



หากต้องการ**ยกเลิกคำสั่งรัน** Application **ให้กดปุ่ม** Ctrl + C

อีกคำสั่งในการรัน Application ที่ไป Browser คือคำสั่ง **ionic serve -l** เป็นคำสั่งสำหรับสั่งรัน ionic lab เราสามารถดู App สาม platform พร้อมกันได้เลยทั้ง Android, iOS และ Windows



การสั่งรัน Application ไปที่ เครื่องทดสอบจริง

สำหรับ Android ใช้คำสั่ง **ionic run android**

สำหรับ iOS ใช้คำสั่ง **ionic run ios**

> **Note:** การรันไปเครื่องทดสอบจริง อย่าลืมต่อสาย USB และเปิด Developer Options ในเครื่องด้วย ถ้าเป็น Android ปกติเข้าที่เมนู Setting->About phone->Build Number (กดต่อเนื่อง 7 ครั้ง)->ติ๊กถูกที่ Stay awake->ติ๊กถูกที่ USB Debugging

การสั่งรัน Application ไปที่ Emulator

สำหรับ Android ใช้คำสั่ง **ionic emulate android**

สำหรับ iOS ใช้คำสั่ง **ionic emulate ios**

การอัปเดต Ionic Application

หาก Ionic มีเวอร์ชันใหม่ออกมาเราสามารถอัปเดตเวอร์ชันใหม่ได้ทันที โดยใช้คำสั่ง

npm install -g ionic@beta

หรือ (สำหรับคนใช้ MAC/Linux)

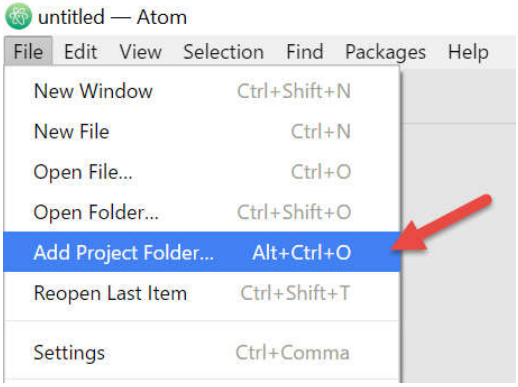
sudo npm install -g ionic@beta

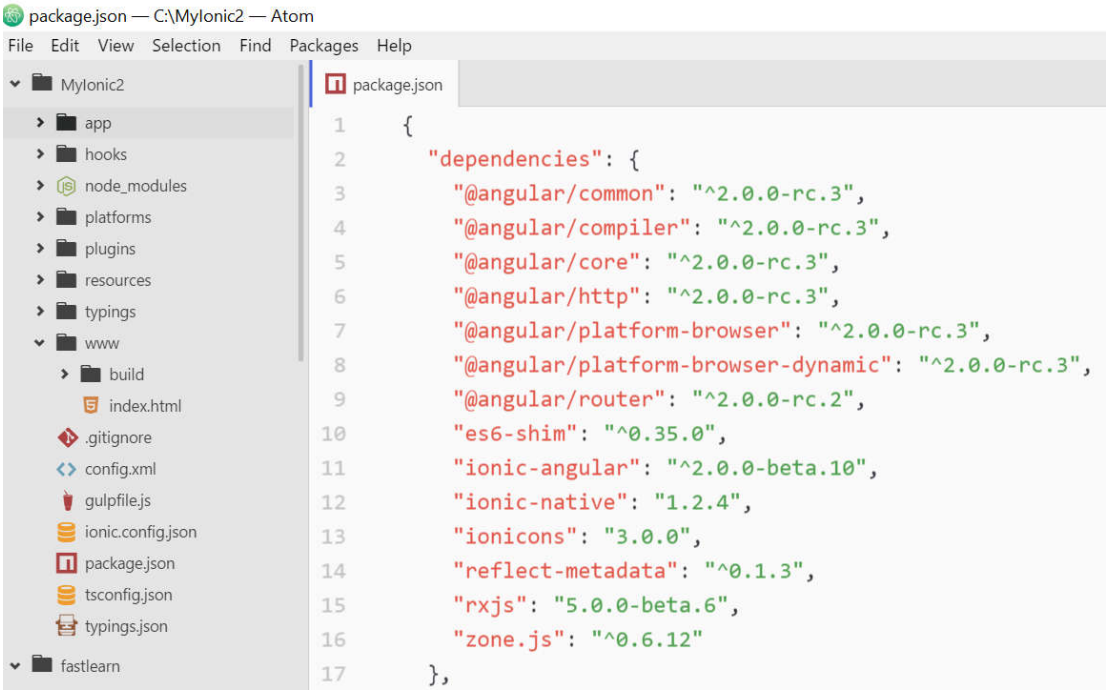
อีกวิธีคือ แก้ไขเลขเวอร์ชันต่างๆ ด้วยตัวเองในไฟล์ **package.json** เสร็จแล้วบันทึกไฟล์ แล้วสั่ง **npm install** ตัว npm จะอัปเดตเวอร์ชันให้เราอัตโนมัติ

> **Note:** การอัปเดต Ionic ใหม่ แนะนำให้เข้าไปอ่าน Change log ของ Ionic ด้วยครับ

การเปิดโปรเจค และเพิ่มเข้าใน Atom Editor

เปิด Atom Editor คลิกเมนู File -> Add Project Folder… แล้วเลือก โฟลเดอร์โปรเจคของเราได้เลยครับ





หากมาถึงขั้นตอนนี้แสดงว่า พร้อมพัฒนา App เรียบร้อยครับ!

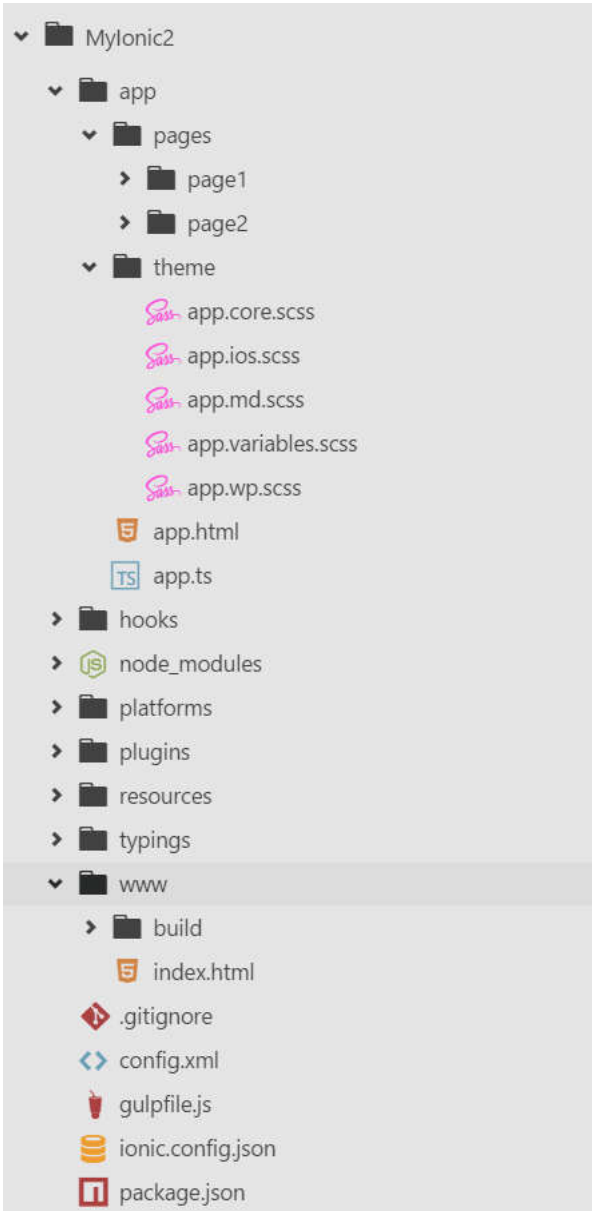
บทที่ 4 แนะนำ Ionic Framework 2

ก่อนจะเข้าสู่การพัฒนา Mobile App ขอเกริ่นนำ Ionic เล็กน้อย และมาทำความรู้จักกับโครงสร้างโฟลเดอร์ต่างๆใน Ionic 2 กันก่อน

อย่างที่เกริ่นไปแล้ว Ionic เป็นรูปแบบการพัฒนา Mobile App ที่เรียกว่า Hybrid App Development คือนักพัฒนาสามารถใช้ความรู้ด้าน Web Application ได้แก่ HTML5, CSS (Sass), JavaScript (JS ES6, Angular 2) มาพัฒนาเป็น Mobile App ได้

โครงสร้างไฟล์ และโฟลเดอร์ต่างๆในโปรเจค

หลังจากติดตั้ง Ionic 2 เรียบร้อยมาดูกันครับว่า โฟลเดอร์สำคัญๆในโปรเจคของเรามีอะไรบ้าง



1. โฟลเดอร์ **app** (default application) เป็นโฟลเดอร์ที่เราใช้งานบ่อยที่สุด ภายในโฟลเดอร์ app ประกอบด้วย
    - โฟลเดอร์ pages มีหน้าที่เก็บโฟลเดอร์หน้าเพจทั้งหมดใน app ของเรา เรียกว่า page component ในแต่ละเพจจะประกอบด้วยไฟล์นามสกุล .html, .scss (ไว้ตกแต่งเพจ) และ .ts (ไว้เขียน logic ต่างๆ) หน้าเพจนี้สามารถ gen code ได้โดยการใช้ Ionic CLI
    - โฟลเดอร์ theme จะเก็บไฟล์นามสกุล .scss (Sass) มีไว้สำหรับแก้ไข และตกแต่งสไตล์ทั้งหมดใน App ของเรา โดยจะแยกเป็น Platforms ได้แก่ iOS (ap.ios.scss), Android (app.md.scss) และ Windows (app.wp.scss)
    - ไฟล์ app.ts เป็น root component หรือเป็นจุดเริ่มของ application เรานั่นเอง
2. โฟลเดอร์ **www** เป็น web root นั่นเอง ห้ามแก้ไขไฟล์ต่างๆในโฟลเดอร์ build แต่สามารถแก้ไข index.html ได้ (หากจำเป็น) และเราสามารถสร้างโฟลเดอร์รูปภาพเพื่อลิงก์แสดงผลได้เหมือนกันเขียน web app ปกติ เช่น <img src="images/pic.jpg">
3. ไฟล์ **config.xml** เป็นไฟล์ตั้งค่าต่างๆเพื่อบอก Cordova ใช้ในการ build ไปยัง Platforms ต่างๆ
4. ไฟล์ **package.json** เป็นไฟล์ตั้งค่าของ npm ใช้สำหรับอัปเดตเวอร์ชัน Ionic และ dependencies ต่างๆ
5. โฟลเดอร์ **resources** มีไว้เก็บ splash screens และ icons ต่างๆใน App ของเรา และใช้ตอน build app การสร้าง resources นั้น สามารถใช้ Ionic CLI ได้
6. โฟลเดอร์ **hooks** เป็นโฟลเดอร์ที่เก็บไฟล์เกี่ยวกับกระบวนการ build ต่างๆ โดยเราสามารถเพิ่ม custom script ในกระบวนการ build ได้ สำหรับมือใหม่ยังไม่แนะนำให้แก้ไขในส่วนนี้
7. โฟลเดอร์ **node_modules** เป็นโฟลเดอร์ไว้เก็บ libraries ต่างๆที่ต้องใช้ในโปรเจคของเรารวมถึง Angular 2 กับ Ionic ด้วย

การใช้ IONIC CLI COMMANDS เพื่อสร้างโค้ดอัตโนมัติ

โดยส่วนใหญ่แล้วเราจะไม่ค่อยได้สร้างไฟล์ต่างๆเองเพราะมี Ionic CLI ช่วยในการ gen โค้ดต่างๆให้ (สะดวกมาก) ในหัวข้อนี้จะมาเรียนรู้วิธีการใช้ Ionic CLI กันครับ รูปแบบคำสั่ง คือ

```
ionic <command> <option>
```

รูปแบบคำสั่งจะขึ้นต้นด้วย ionic ตามด้วยคำสั่งที่เราต้องการ หากมี option ของคำสั่งนั้นก็สามารถใส่เข้าไปได้ เช่น ionic serve -l
มีหลายคำสั่งที่เราใช้กันไปบ้างแล้ว แต่ใช้บ่อยมาก จะเป็นคำสั่งที่เรียกว่า Generate command เราสามารถ gen code ต่างๆแบบอัตโนมัติไม่ต้องพิมพ์เองทั้งหมด ได้แก่ การสร้าง page, component, directive, pipe, provider และ tabs รูปแบบคำสั่งมีดังนี้

```
ionic g [page|component|directive|pipe|provider|tabs]
```

ยกตัวอย่างเช่น หากต้องการสร้างหน้าเพจใหม่ก็ใช้คำสั่ง **ionic g page product**

Note: หากต้องการดูคำสั่งเกี่ยวกับ generate ทั้งหมดให้พิมพ์ ionic g --list และเราสามารถใช้ generate แทน g ได้

บทที่ 5 การสร้าง Pages, Basic Navigation และ Passing Data

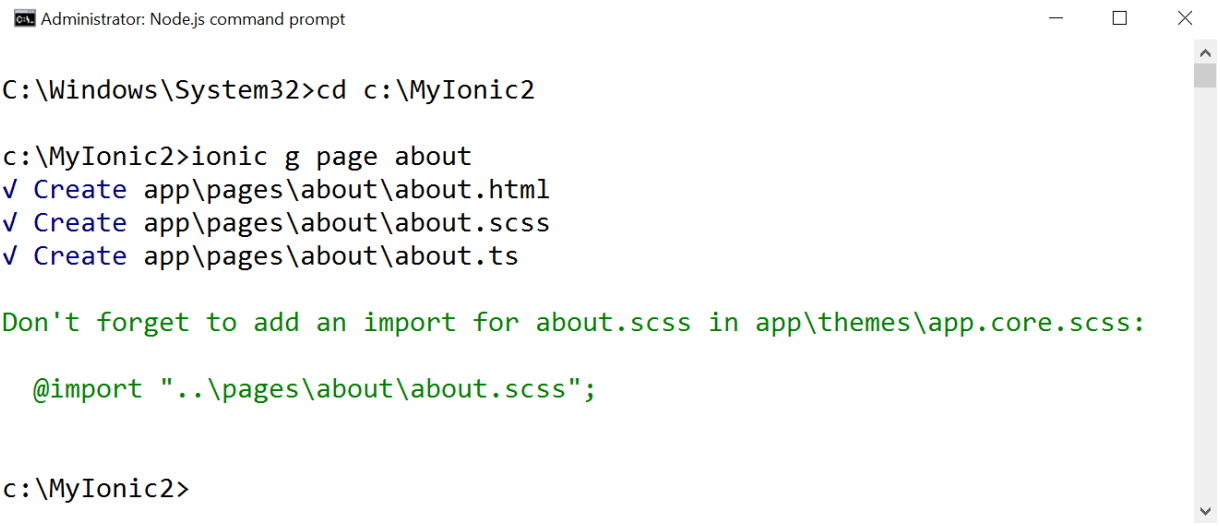
การสร้าง Pages

การสร้างหน้าเพจขึ้นมาใหม่ เราจะใช้ Ionic CLI ด้วยคำสั่ง ionic g page <**ชื่อเพจ**> --ts

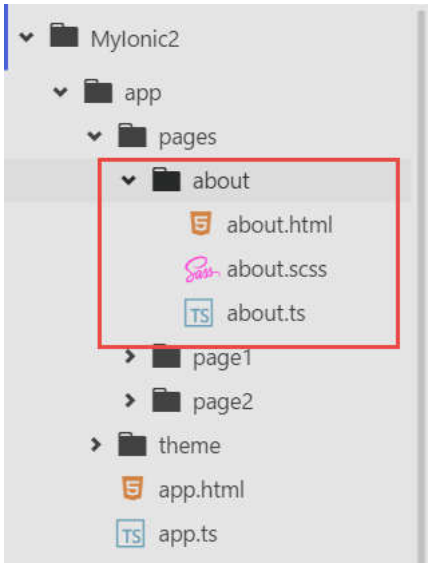
ทดลองสร้างเพจชื่อว่า About

การสร้างเพจใหม่มีขั้นตอน ดังนี้

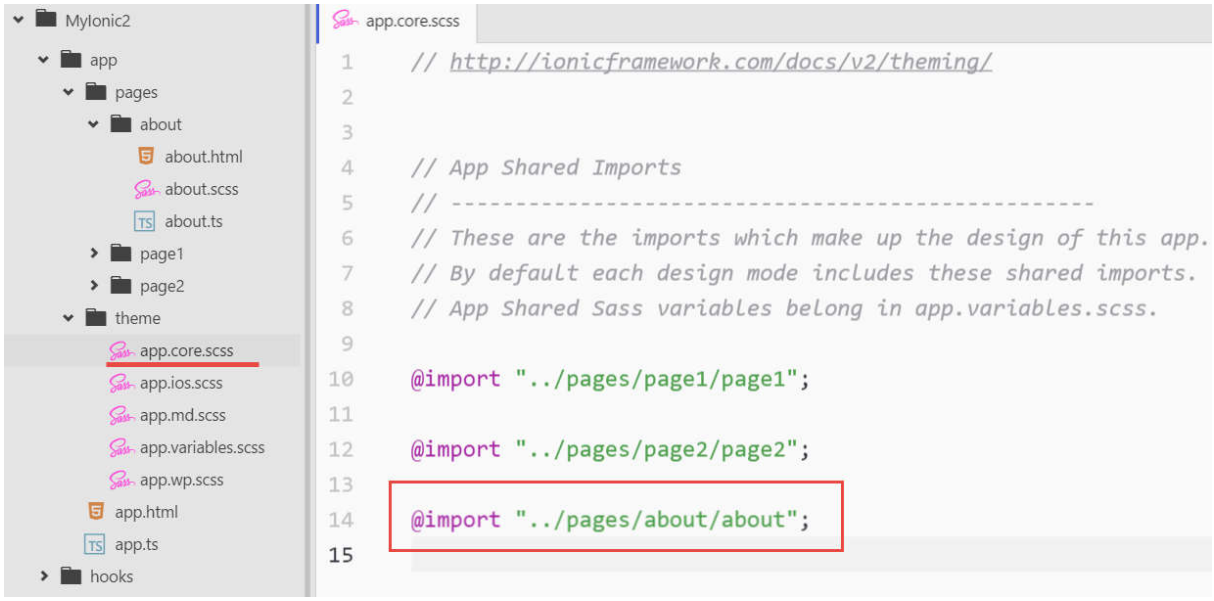
1. พิมพ์คำสั่ง ionic g page about –ts แล้วกด Enter



2. เสร็จแล้ว Ionic CLI จะ gen โค้ดและสร้างไฟล์ให้เราอัตโนมัติ



3. เปิดไฟล์ app\themes\app.core.scss เพิ่มโค้ด @import "../pages/about/about"; เข้าไป



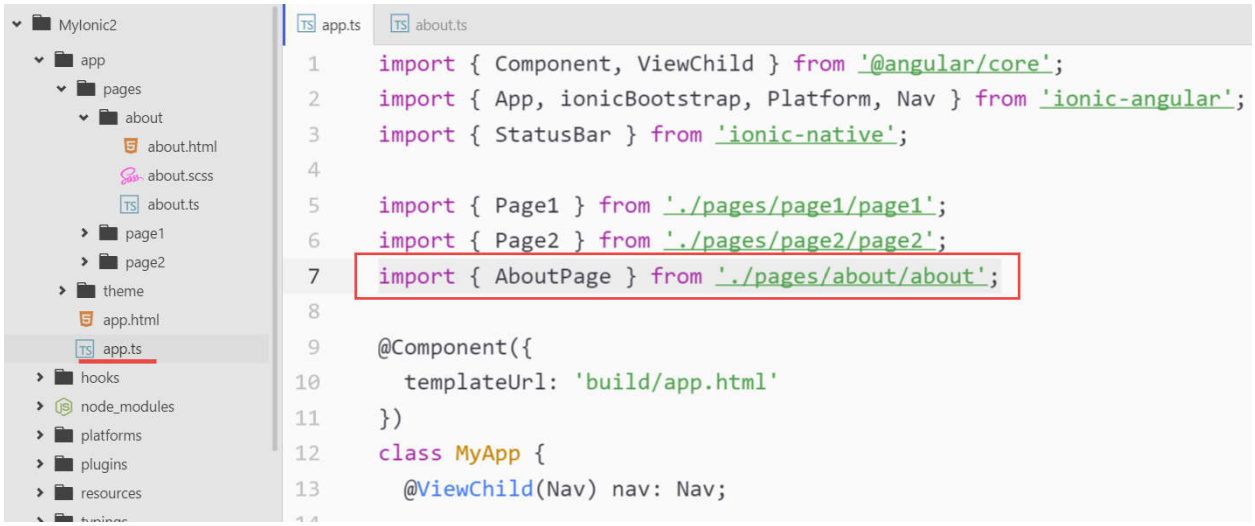
4. เสร็จเรียบร้อยสำหรับการสร้างเพจ อย่าลืมบันทึกไฟล์เป็นระยะนะครับ

Note: ต่อไปหากมีการสร้าง Page ก็ให้ทำตามขั้นตอนด้านบนนี้ได้เลย

ทดลองสร้างเมนู “เกี่ยวกับเรา” เพื่อเปิดหน้าเพจ About

หลังจากทดลองสร้างเพจ About เรียบร้อย มาลองสร้างเมนู (sidebar) เพื่อเปิดหน้าเพจ About ดูครับ

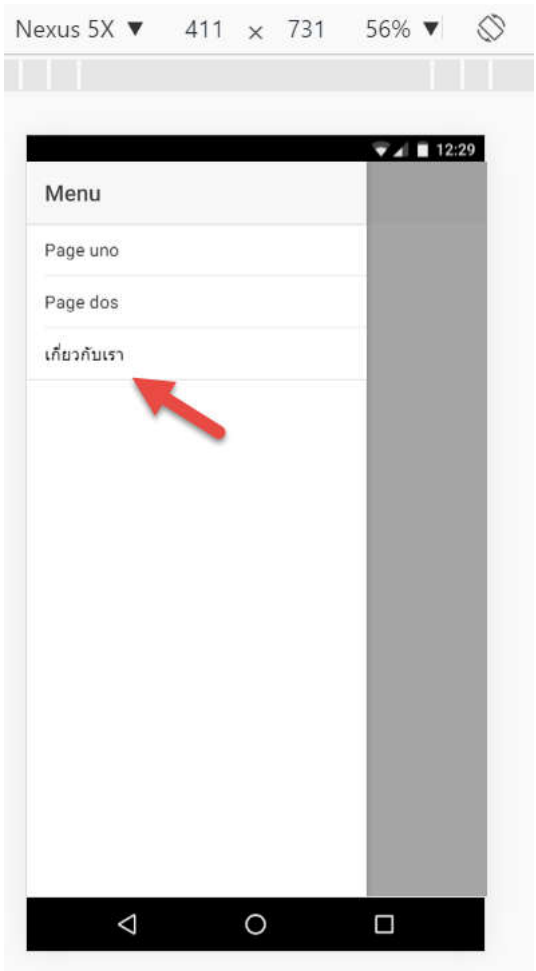
1. เปิดไฟล์ app\app.ts จากนั้นทำการ import AboutPage เข้ามาด้วยคำสั่ง import { AboutPage } from './pages/about/about';



Note: หลังคำสั่ง from ให้ระบุ path ไฟล์ (about.ts) ให้ถูกต้อง

2. เพิ่มโค้ดสมาชิก array อีก 1 บรรทัด **{ title: 'เกี่ยวกับเรา', component: AboutPage }** อย่าลืมเติมคอมม่าต่อตัวสุดท้ายด้วย สามารถเปลี่ยน title ได้ตามต้องการ

3. ทดลองรันด้วยคำสั่ง ionic serve



Note: การรันคำสั่ง ionic serve แนะนำให้รันทิ้งไว้เลยโดยให้เปิด Node.js command prompt อีก 1 หน้าต่าง

4. หากต้องการให้หน้า About มี**เมนู**ด้วย ให้แก้ไฟล์ about.html ดังนี้

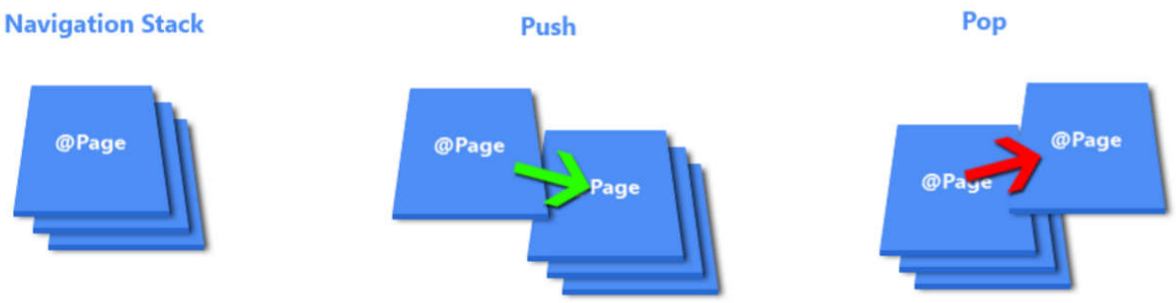
```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
 <ion-title>เกี่ยวกับเรา</ion-title>
</ion-navbar>
</ion-header>

<ion-content class="about">
 ยินดีต้อนรับทุกท่าน
</ion-content>
```

บันทึกไฟล์ แล้ว Refresh browser อีกครั้ง

Note: ต่อไปหากมีการสร้างเพจใหม่ และอยากเพิ่มเมนูให้กลับมาทบทวนในหัวข้อนี้

พื้นฐาน Navigation

เมื่อเราสร้างเพจได้แล้ว ลำดับต่อไปจะต้องทำความเข้าใจเกี่ยวกับ Navigation ครับ หลักการของ Navigation จะอยู่ในรูปแบบ **Stack** คือมีการนำเพจเข้ามาแสดง เรียกว่า **push** และการนำเพจออกไป เรียกว่า **pop** นั่นเอง



Note: @Page ในรูปให้มองว่าเป็น @Component ในแต่ละเพจ

การที่เราจะทำ Navigation ได้จะต้อง import **NavController** เข้ามาก่อน และสร้างตัวแปรส่งเข้าไปยัง constructor ซึ่งปกติหากเราสร้างเพจด้วย Ionic CLI จะมีโค้ดส่วนนี้ให้อยู่แล้ว

```
1  import { Component } from '@angular/core';
2  import { NavController } from 'ionic-angular';
3
4  @Component({
5    templateUrl: 'build/pages/about/about.html',
6  })
7  export class AboutPage {
8    constructor(private nav: NavController) {}
9  }
```

การ Push หน้าเพจขึ้นมาแสดง

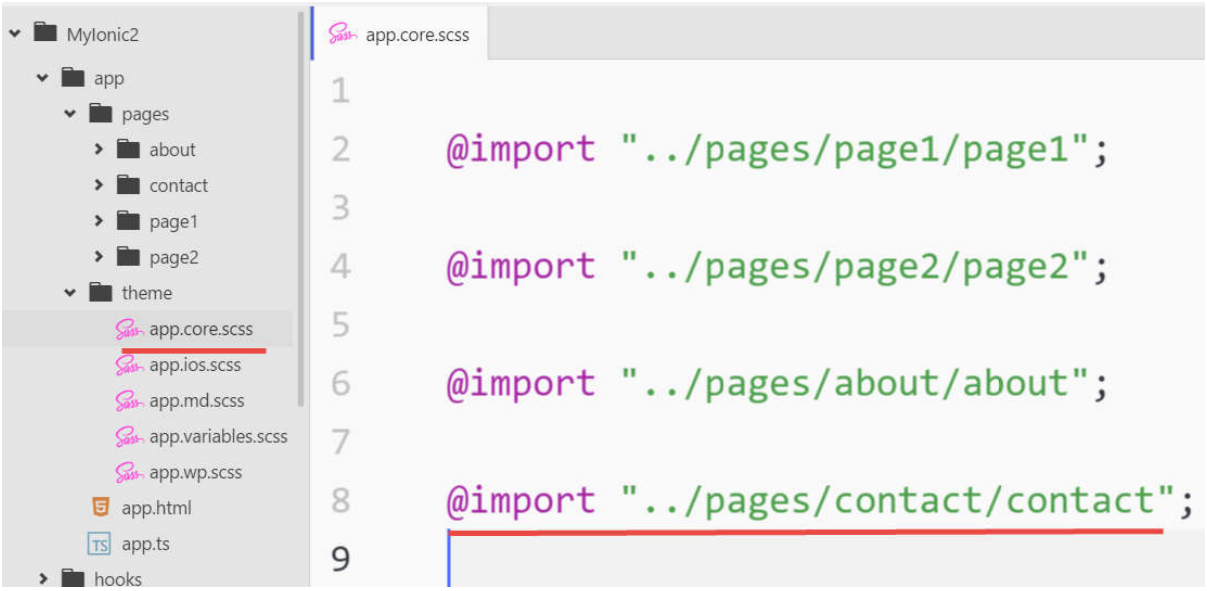
เพื่อการยกตัวอย่างให้เห็นภาพมากขึ้น เราจะสร้างปุ่มที่เพจ about เมื่อคลิกปุ่มแล้วให้เปิด (push) หน้าเพจ contact ครับ

ขั้นตอน มีดังนี้

1. สร้างเพจ contact ก่อนโดยใช้คำสั่ง ionic g page contact --ts

```
Administrator: Node.js command prompt

c:\MyIonic2>ionic g page contact --ts
√ Create app\pages\contact\contact.html
√ Create app\pages\contact\contact.scss
√ Create app\pages\contact\contact.ts

Don't forget to add an import for contact.scss in app\themes\app.core.scss:

  @import "..\pages\contact\contact.scss";
```

เปิดไฟล์ app\theme\app.core.scss เพิ่มโค้ด @import "../pages/contact/contact";



2. เปิดไฟล์ app\pages\contact\contact.html แทรกโค้ดเพื่อเพิ่มปุ่ม menuToggle ดังนี้

```
<ion-header>
<ion-navbar>
 <button menuToggle>
   <ion-icon name="menu"></ion-icon>
 </button>
 <ion-title>ติดต่อเรา</ion-title>
</ion-navbar>
</ion-header>

<ion-content  class="contact">
Codingthailand.com
</ion-content>
```

3. เปิดไฟล์ **app\pages\about\about.html** เพื่อเพิ่มปุ่ม, ไอคอน และเหตุการณ์ (event) คลิก พร้อมตั้งชื่อฟังก์ชันชื่อว่า goToContact()

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>เกี่ยวกับเรา</ion-title>
</ion-navbar>
</ion-header>

<ion-content  class="about">
 ยินดีต้อนรับทุกท่าน
<br>
  <button (click)="goToContact()">
    <ion-icon name="call"></ion-icon>
    เปิดหน้าติดต่อเรา
  </button>
</ion-content>
```

4. เปิดไฟล์ **app\pages\about\about.ts** เพื่อ import เพจ contact เข้ามา แล้วเขียน method เพื่อเปิด (push) เพจ contact ดังนี้

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import {ContactPage} from '../contact/contact';

@Component({
  templateUrl: 'build/pages/about/about.html',
})
export class AboutPage {
```

```
  constructor(private nav: NavController) {
    this.nav = nav;
  }

  goToContact() {
    this.nav.push(ContactPage);
  }
}
```

อธิบายโค้ด เราได้ Import คลาส ContactPage เข้ามา และใน constructor เราได้กำหนดตัวแปร nav ให้กับ this.nav คือ กำหนดตัวแปร nav ให้กับตัวแปร nav **ของคลาสนี้**ครับ จากนั้นเราได้สร้างฟังก์ชันหรือ method ชื่อว่า goToContact() แล้วเรียกใช้ this.nav.**push**(ContactPage) เพื่อเปิดเพจ Contact

about.html

```
<ion-navbar *navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>เกี่ยวกับเรา</ion-title>
</ion-navbar>

<ion-content padding class="about">
  ยินดีต้อนรับทุกท่าน
  <br>
  <button (click)="goToContact()">
    <ion-icon name="call"></ion-icon>
    เปิดหน้าติดต่อเรา
  </button>
</ion-content>
```

about.ts

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import {ContactPage} from '../contact/contact';

@Component({
  templateUrl: 'build/pages/about/about.html',
})
export class AboutPage {
  constructor(private nav: NavController) {
    this.nav = nav;
  }

  goToContact() {
    this.nav.push(ContactPage);
  }
}
```

Note: หากต้องการ push หน้าไหนก็ให้ import และใส่ชื่อคลาสให้กับ argument ของ push นั่นเอง

การ Pop หน้าเพจ

หลังจากที่เราได้เปิดหน้าเพจ Contact แล้วมาลอง pop ออกกันครับ จริงๆแล้วคำสั่งในการ pop ก็ใช้งานคล้ายกันกับ push ดังนี้ this.nav.**pop()**; ตัวอย่าง

```
class SecondView {

  constructor(nav:NavController) {
```

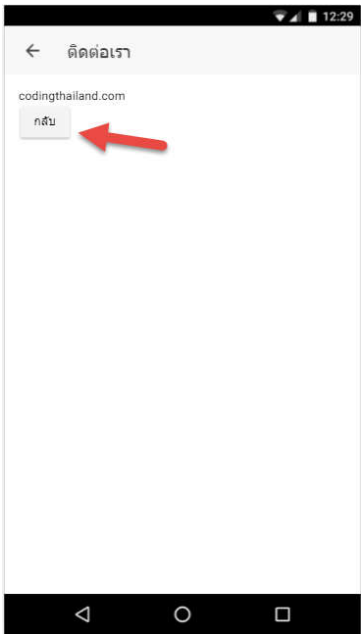
```
    this.nav = nav;
  }
  goBack(){
    this.nav.pop();
  }
}
```

แต่มีอีกวิธีหากไม่ต้องการสร้างฟังก์ชัน หรือ Method เพิ่ม เราสามารถระบุ attribute หรือ directive ชื่อว่า nav-pop เข้าไปที่ปุ่มได้เลย มาลองเขียนกันดูครับ

1. เปิดไฟล์ **app\pages\contact\contact.html** แทรกโค้ด ดังนี้

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>ติดต่อเรา</ion-title>
</ion-navbar>
</ion-header>

<ion-content  class="contact">
  codingthailand.com <br>
  <button light nav-pop>กลับ</button>
</ion-content>
```

2. บันทึกไฟล์ แล้วลองคลิกปุ่ม “เปิดหน้าติดต่อเรา” ที่เพจ About จากนั้นคลิกปุ่ม “กลับ” แค่นี้เราก็ pop เพจได้เรียบร้อย



การกำหนดหน้าเพจ ให้เป็น root page

จะสังเกตว่าเวลาที่เรา push หน้าเพจไหนก็ตาม เพจนั้นจะมาทับอยู่บนเพจเดิมที่มีอยู่ และตัวมันเองสามารถ pop ได้ แต่ถ้าหากเราต้องการเปิดเพจใดๆ แล้วอยากให้เป็น root page (เพจหลัก) ให้ใช้คำสั่ง this.nav.**setRoot**(ชื่อคลาสเพจ);

ให้ทุกคนลองแก้ไขไฟล์ **app\pages\about\about.ts** ที่ method **goToContact()**แล้วลองรันดูอีกครั้ง

```
goToContact() {
   //this.nav.push(ContactPage);
   this.nav.setRoot(ContactPage);
 }
```

การส่งข้อมูลระหว่างเพจ (Passing Data between Pages)

หัวข้อนี้เป็นหัวข้อที่สำคัญมาก เพราะเราจะได้ใช้บ่อยๆในการพัฒนา Mobile App เช่นในการทำเพจในรูปแบบของ **Master Detail** ยกตัวอย่างเช่น ถ้าเรามีหน้าเพจสินค้า หากคลิกที่ชื่อสินค้า จะมีการส่งรหัสสินค้าไปที่เพจรายละเอียดสินค้าเพื่อดึงข้อมูลจากฐานข้อมูลมาแสดง เป็นต้น

ในการส่งข้อมูลนั้น ที่เพจหลัก เราสามารถส่งข้อมูลไปกับ method push() ได้เลย (หรือ method setRoot() ก็ได้) ในตัวอย่างนี้เราจะส่งข้อมูลจากเพจ About ไปที่เพจ Contact ครับ

1. เปิดไฟล์ app\pages\about\about.ts เราจะทดลองส่งข้อมูล 2 ตัว ได้แก่ myName กับ myEmail โดยแก้ไข method goToContact() ดังนี้

```
goToContact() {
   this.nav.push(ContactPage, {
        myName: 'Akenarin Komkoon',
        myEmail: 'codingthailand.com'
    });
   /*this.nav.setRoot(ContactPage);*/
 }
```

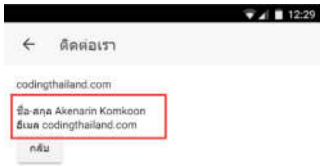
2. เปิดไฟล์ app\pages\contact\contact.ts ซึ่งเราจะใช้เป็นหน้านี้สำหรับรับข้อมูลจากเพจ About ในการรับข้อมูลที่ส่งมาเราจะต้อง import NavParams ก่อน และต้องกำหนดที่ constructor ด้วย ในการเรียกข้อมูลที่รับมานั้นเราจะใช้ method get() ครับ

```
import { Component } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';

@Component({
  templateUrl: 'build/pages/contact/contact.html',
})
export class ContactPage {
  myName: string;
  myEmail: string;
  constructor(private nav: NavController, private navParams: NavParams) {
    this.myName = navParams.get('myName');
    this.myEmail = navParams.get('myEmail');
  }
}
```

เปิดไฟล์ app\pages\contact\contact.html และเพิ่มโค้ดเพื่อแสดงผลตัวแปรที่รับมา ดังนี้

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>ติดต่อเรา</ion-title>
</ion-navbar>
</ion-header>
<ion-content  class="contact">
  codingthailand.com <br>
  <br>
  ชื่อ-สกุล {{ myName }} <br>
  อีเมล {{ myEmail }} <br>

  <button light nav-pop>กลับ</button>
</ion-content>
```

3. ทดสอบ App แล้วลองคลิกปุ่ม “เปิดหน้าติดต่อเรา” ในเพจ About ดูครับ



Note: หากไม่แสดง อาจเขียนโค้ดผิด หรือไม่ถูกต้องทำให้เกิด errors แก้ไข แล้วรัน ionic serve อีกครั้ง มาถึงจุดนี้หากอยากลองแก้ไขข้อความหน้าแรกของ App และเมนูอื่นๆ ให้ทดลองแก้ไขด้วยตัวเองได้เลยครับ

บทที่ 6 Ionic 2 Components

Components คืออะไร

Components คือ ชุดของ User Interface (UI) สำเร็จรูปที่เราสามารถนำมาใช้ได้ทันที เช่น Button, Cards, Alerts, Modal, Lists เป็นต้น โดยใน Component แต่ละตัวจะมีข้อกำหนด ฟังก์ชัน และการใช้งานแตกต่างกัน สามารถดูรายละเอียด และทดลองดูตัวอย่าง Components ทั้งหมดได้ที่ http://ionicframework.com/docs/v2/components/

การใช้ Components เบื้องต้น

เนื่องจาก Component มีหลายตัว ในหนังสือเล่มนี้จะยกตัวอย่างเฉพาะที่ใช้บ่อย แต่ถ้ารู้หลักการแล้วสามารถประยุกต์ใช้ได้กับทุกตัวครับ

Buttons

เป็นปุ่มกด สามารถใช้งานได้ง่ายมาก เพียงแค่คำสั่ง <button>Button</button> หากต้องการปรับสีปุ่มต่างๆก็เขียนโค้ดได้ดังนี้

```
<button light>Light</button>
<button>Primary</button>
<button secondary>Secondary</button>
<button danger>Danger</button>
<button dark>Dark</button>
```

ดูรายละเอียดการใช้งานทั้งหมดได้ที่ http://ionicframework.com/docs/v2/components/#round-buttons

Icons

ใน Ionic 2 จะมีไอคอนสำเร็จรูปต่างๆมาให้เราใช้เรียบร้อย โดยมีการใช้งาน ดังนี้

```
<ion-icon name="heart"></ion-icon>
```

หากอยากเปลี่ยนไอคอนแค่ระบุชื่อ (name) เข้าไปได้ตามต้องการ ดูไอคอนทั้งหมดได้ที่นี่ http://ionicframework.com/docs/v2/ionicons/

Grid

เราสามารถแบ่งหน้าเพจในรูปแบบของคอลัมน์ได้ โดยสามารถระบุได้ว่าแต่ละคอลัมน์จะให้กว้างเท่าไหร่ หากใครเคยใช้ CSS Framework เช่น Bootstrap 3 จะมีแนวคิดการใช้แบบเดียวกันครับ รายละเอียดการใช้ดูได้ที่ http://ionicframework.com/docs/v2/components/#grid

มาลองสร้างเมนูหน้าแรกด้วย Grid กันครับ

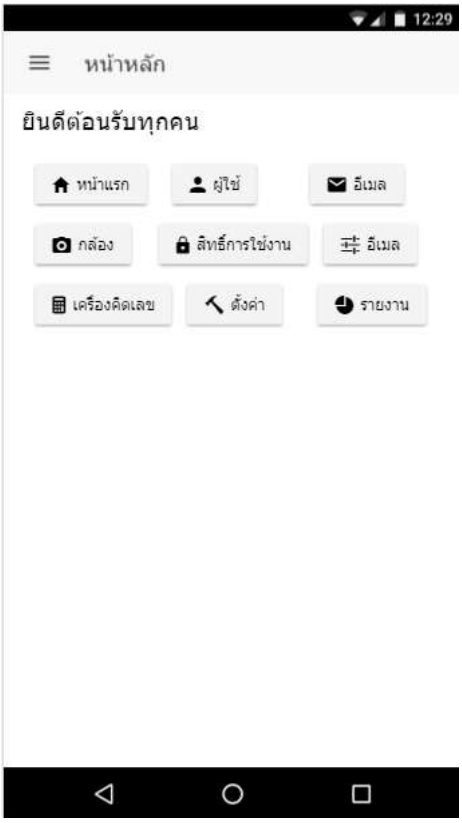
1. เปิดไฟล์ app\pages\page1\page1.html ให้แก้ไขโค้ดหน้าเพจ ดังนี้

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>หน้าหลัก</ion-title>
</ion-navbar>
</ion-header>

<ion-content  class="page1">
  <h3>ยินดีต้อนรับทุกคน</h3>
  <ion-grid>
    <ion-row>
        <ion-col width-33>
            <a button light>
                <ion-icon name='home'></ion-icon>
                หน้าแรก
            </a>
        </ion-col>
        <ion-col width-33>
            <a button light>
                <ion-icon name='person'></ion-icon>
                ผู้ใช้
            </a>
        </ion-col>
```

```
    <ion-col width-33>
        <a button light>
            <ion-icon name='mail'></ion-icon>
            อีเมล
        </a>
    </ion-col>
</ion-row>

<ion-row>
    <ion-col>
        <a button light>
            <ion-icon name='camera'></ion-icon>
            กล้อง
        </a>
    </ion-col>
    <ion-col>
        <a button light>
            <ion-icon name='lock'></ion-icon>
            สิทธิ์การใช้งาน
        </a>
    </ion-col>
    <ion-col>
        <a button light>
            <ion-icon name='options'></ion-icon>
            อีเมล
        </a>
    </ion-col>
</ion-row>

<ion-row>
    <ion-col>
        <a button light>
            <ion-icon name='calculator'></ion-icon>
            เครื่องคิดเลข
        </a>
```

```
            </ion-col>
            <ion-col>
                <a button light>
                    <ion-icon name='hammer'></ion-icon>
                    ตั้งค่า
                </a>
            </ion-col>
            <ion-col>
                <a button light>
                    <ion-icon name='pie'></ion-icon>
                    รายงาน
                </a>
            </ion-col>
        </ion-row>
    </ion-grid>
</ion-content>
```

2. บันทึกไฟล์ แล้วรันดูอีกครั้ง



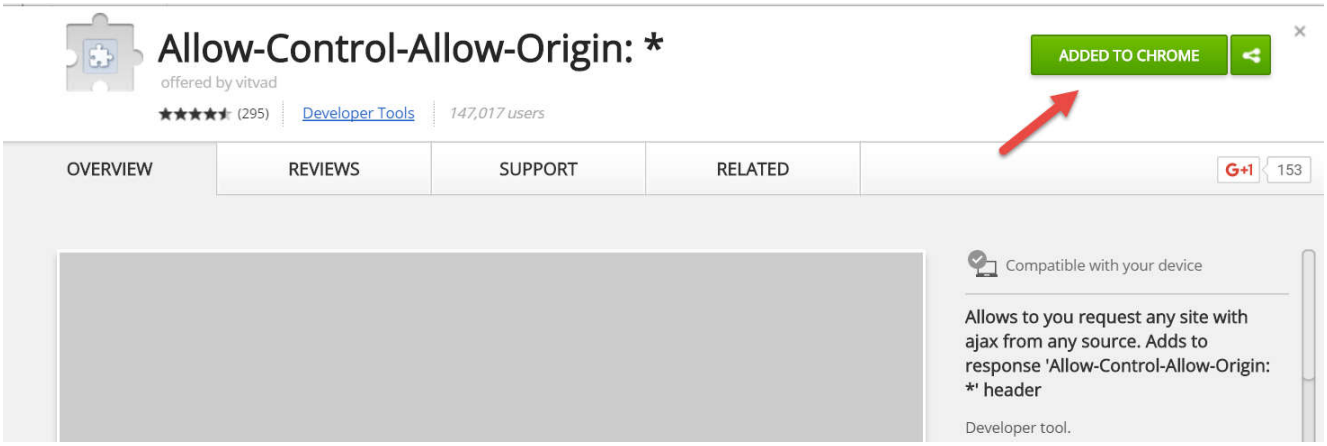
Components อื่นๆ มักใช้กับฟอร์ม หรือเกี่ยวข้องกับ Data ต่างๆ จะมีแทรกในบทต่อๆไปอีกครั้ง

บทที่ 7 การเรียกใช้ API ด้วย Angular 2 Services

ในการเขียน Mobile App บางครั้งเราจะต้องติดต่อกับฐานข้อมูล หรือติดต่อกับระบบ Backend ต่างๆ อาจมีการเรียกดูข้อมูลเพื่อนำมาแสดงผล การเพิ่มข้อมูล แก้ไขข้อมูล หรือลบข้อมูลก็ได้ สำหรับ Ionic 2 จะใช้ความสามารถของ **Angular 2 Services** เพื่อช่วยจัดการกับข้อมูลต่างๆในฝั่ง Backend ครับ

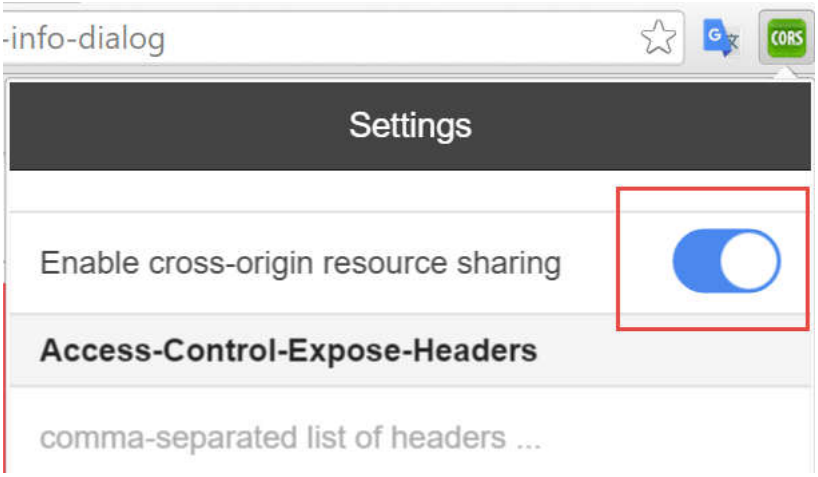
> **Note:** ดูรายละเอียดเชิงลึกของ Angular 2 Services ได้ที่ https://angular.io/docs/ts/latest/guide/server-communication.html

ลำดับแรกให้เราทำการติดตั้ง Google chrome extension ก่อน คลิกเพื่อติดตั้งได้ที่ https://goo.gl/nFxkcr



Extension ตัวนี้จะทำหน้าที่อนุญาตให้เราสามารถรับส่งข้อมูลกับ Server ได้ เพราะฉะนั้นแนะนำให้ติดตั้งทุกคนครับ หากไม่ติดตั้งเราจะไม่สามารถทดสอบ App ของเราได้โดยจะขึ้น errors เกี่ยวกับ Allow Control Allow Origin นั่นเอง

หลังจากติดตั้งแล้วหากเรากำลังพัฒนา App อยู่ก็แนะนำให้เปิดไว้ครับ (จะเป็นสีเขียว) หากไม่ได้ใช้ก็ปิดไว้เลยครับ



ขั้นตอนการใช้ Angular 2 Services หรือ HTTP Services ใน Ionic 2

การใช้ HTTP Services นั้นมีวิธีการใช้อยู่ 2 วิธี คือ

1. **เขียนเอง** (เรียก method get()) ต้อง import http เข้ามาก่อน และเขียนจัดการข้อมูล
2. **ใช้ผ่าน providers** (Promise) ซึ่ง Ionic 2 มีส่วนในการ gen code ให้อยู่แล้ว

Note: การใช้ HTTP Services นั้นไม่จำเป็นต้องจัดการข้อมูลฝั่ง Backend อย่างเดียว สามารถจัดการไฟล์ JSON ได้ด้วย

ส่วนใหญ่แล้วในฝั่ง Backend จะคืนค่าข้อมูลอยู่ในรูปแบบ JSON มาทดลองเขียนกันเลยครับ

1. **การใช้ HTTP Services (เขียนเอง)** มีขั้นตอนหลักๆ ดังนี้
    1.1 import HTTP_PROVIDERS, Http เข้ามาใช้ในเพจที่ต้องการ
    `import {HTTP_PROVIDERS, Http} from 'angular2/http';`
    1.2 กำหนด **providers: [HTTP_PROVIDERS]** ให้กับ @Component นั้นๆ
    1.3 ประกาศ**ตัวแปร HTTP ที่** constructor ของคลาสนั้นๆ

มาทดลองดึงข้อมูลจาก Server กันครับ ในตัวอย่างนี้ผมจะทดลองดึงข้อมูลผ่าน URL นี้

http://codingthailand.com/api/get_courses.php

ไฟล์ get_courses.php นี้จะทำดึงข้อมูลรายชื่อคอร์สเรียนฟรีจากเว็บ codingthailand.com และคืนค่ามาในรูปแบบ JSON ตัวอย่างการเขียน php มีดังนี้

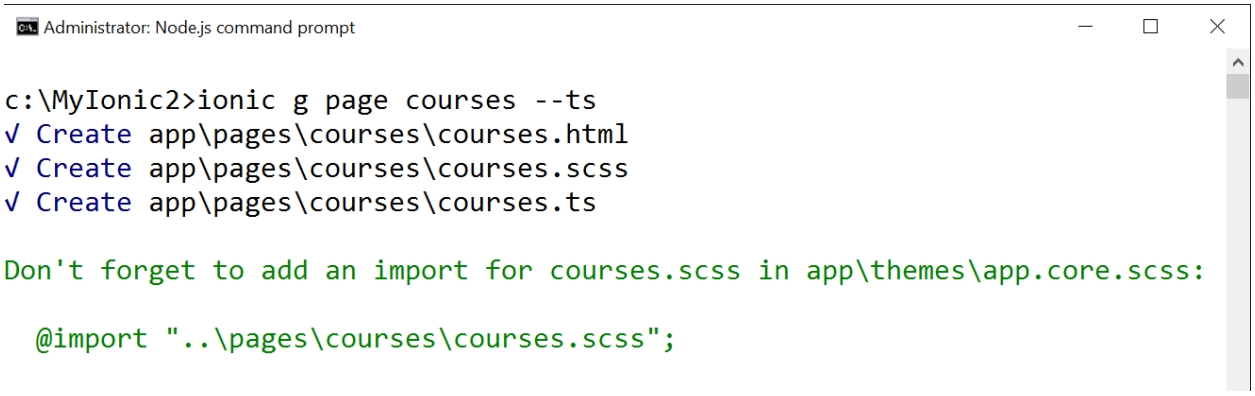
```
<?php
        $link = mysqli_connect('localhost', 'root', '', 'mydb');
        mysqli_set_charset($link, 'utf8');
        $sql = "SELECT * FROM courses";
        $result = mysqli_query($link, $sql);
        $coursesArray = array();
        while ($row = mysqli_fetch_assoc($result)) {
                $coursesArray[] = $row;
        }
        echo json_encode($coursesArray);
```

ทดลองรัน http://codingthailand.com/api/get_courses.php ที่ Browser จะได้ผลลัพธ์ดังนี้

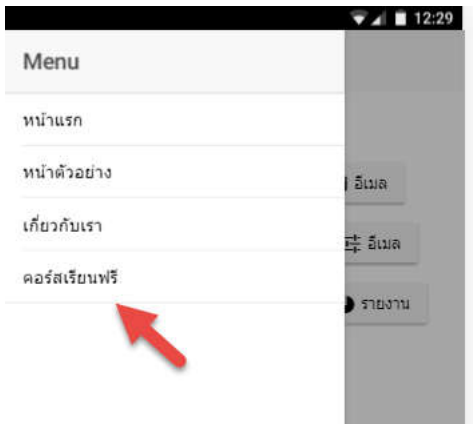
จากภาพ เราจะได้ข้อมูลจากฐานข้อมูลมา โดยแต่ละแถวจะมีคอลัมน์ และค่าข้อมูลแสดงออกมา ในฝั่งของ Ionic เวลาอ้างถึงคอลัมน์ในฐานข้อมูลก็อย่าลืมเขียนและอ้างชื่อให้ตรงกันด้วยนะครับ

ในหัวข้อนี้เราจะลองดึงข้อมูลมาแสดงในรูปแบบ Lists มีขั้นตอน ดังนี้

1. สร้างเพจชื่อว่า courses ใช้คำสั่ง ionic g page courses --ts แล้วกด Enter และแน่นอนหลังทุกครั้งหลังสร้าง Page อย่าลืมเปิดไฟล์ app\theme\app.core.scss แล้วเพิ่มคำสั่ง @import "../pages/courses/courses"; ด้วย



2. สร้าง side menu ตามนี้ (ให้ทดลองเขียนเพิ่มเมนูด้วยตัวเอง)



3. เปิดไฟล์ app\pages\courses\courses.ts เพิ่มโค้ด ดังนี้

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import { HTTP_PROVIDERS, Http } from '@angular/http'; //นำเข้า HTTP_PROVIDERS, Http เข้ามาใช้งาน

@Component({
  providers: [HTTP_PROVIDERS],
  templateUrl: 'build/pages/courses/courses.html',
})
export class CoursesPage {
   items: Array<{}>; //ประกาศตัวแปร array เปล่า
  constructor(private nav: NavController, private http: Http) { //ประกาศตัวแปร http
    this.http = http; //นำ http มากำหนดให้ http ของคลาส
    //ใช้ method get() เพื่อดึงข้อมูล
    //ใช้ method subscribe() เพื่อดึงค่า json object เข้ามาใช้งานใน component
    this.http.get("http://codingthailand.com/api/get_courses.php").subscribe(
       (response) => {
            this.items = response.json(); //ดึงข้อมูลที่ได้ แล้วกำหนดให้กับตัวแปร itmes ของคลาสนี้
         }
        );
     }
}
```

4. เมื่อตัวแปร items ได้รับข้อมูลจาก Server เรียบร้อย ต่อไปมาลองแสดงผลกัน โดยการแสดงผลนั้นเราจะใช้ Lists Component (<ion-list>) ในการแสดง ให้ เปิดไฟล์ app\pages\courses\courses.html เขียนโค้ดดังนี้

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
```

```
  <ion-title>รายการคอร์สเรียนฟรี</ion-title>
</ion-navbar>
</ion-header>

<ion-content  class="courses">
  <ion-content>
    <ion-list>
      <button ion-item *ngFor="let item of items">
        <h2>{{item.c_detail}}</h2>
        <p>{{item.c_title}}</p>
      </button>
    </ion-list>
</ion-content>
```

อธิบายโค้ด ในโค้ดเราจะครอบด้วย <ion-list> แต่ละรายการของ Lists ให้ระบุ ion-item เราใช้ *ngFor เพื่อวนลูปตัวแปร items ที่ซึ่งอยู่ใน courses.ts นั่นเอง

courses.ts

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import { HTTP_PROVIDERS, Http } from '@angular/http'; //

@Component({
  providers: [HTTP_PROVIDERS],
  templateUrl: 'build/pages/courses/courses.html',
})
export class CoursesPage {
  items: Array<{}>; //ประกาศตัวแปร array เปล่า
  constructor(private nav: NavController, private http:
    this.http = http; //นำ http มากำหนดให้ http ของคลาส
    //ใช้ method get() เพื่อดึงข้อมูล
    //ใช้ subscribe() เพื่อดึงค่า object เข้ามาใช้งานใน componer
    this.http.get("http://codingthailand.com/api/get_cou
      (response) => {
            this.items = response.json(); //ดึงข้อมูลที่ได้ u
        }
      );
    }
}
```

courses.html

```
<ion-navbar *navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>รายการคอร์สเรียนฟรี</ion-title>
</ion-navbar>

<ion-content padding class="courses">
  <ion-content>
    <ion-list>
      <button ion-item *ngFor="let item of items"
        <h2>{{item.c_detail}}</h2>
        <p>{{item.c_title}}</p>
      </button>
    </ion-list>
</ion-content>
```

5. เสร็จแล้ว บันทึกไฟล์ แล้วลองรัน หรือ refresh browser ดูอีกครั้ง

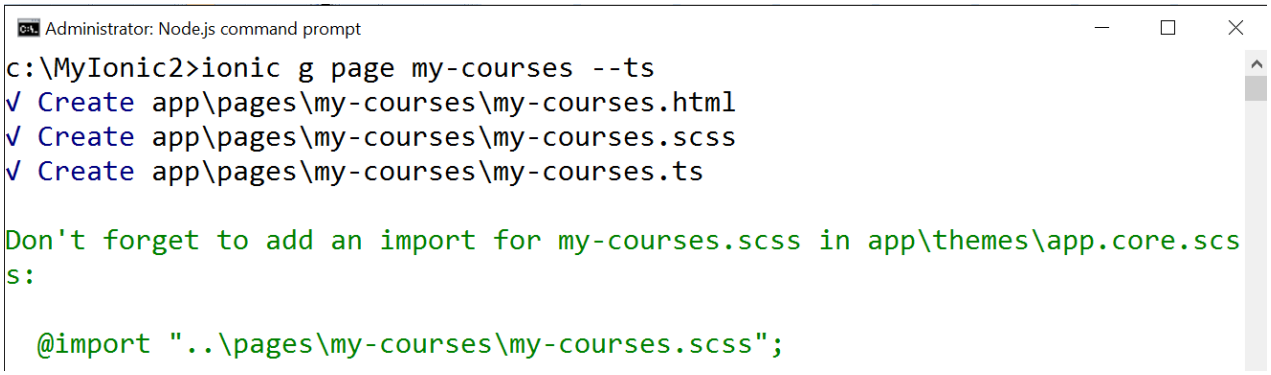
2. **การใช้** HTTP Services **ผ่าน** providers

หลังจากที่เราลองเขียนเองเรียบร้อย ต่อมามาลองเขียนโดยใช้ providers กันครับ ใน Ionic 2 นั้นเราสามารถใช้ Ionic CLI ช่วย gen โค้ด providers ให้เราได้ง่ายมาก มีรูปแบบคำสั่ง ดังนี้

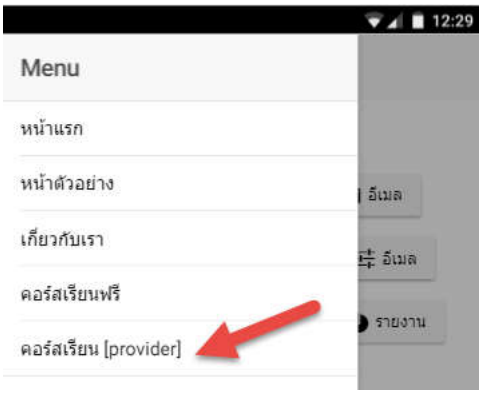
ionic g **provider** <ชื่อ provider> --ts

ขั้นตอนการจัดการข้อมูลด้วย providers มีดังนี้

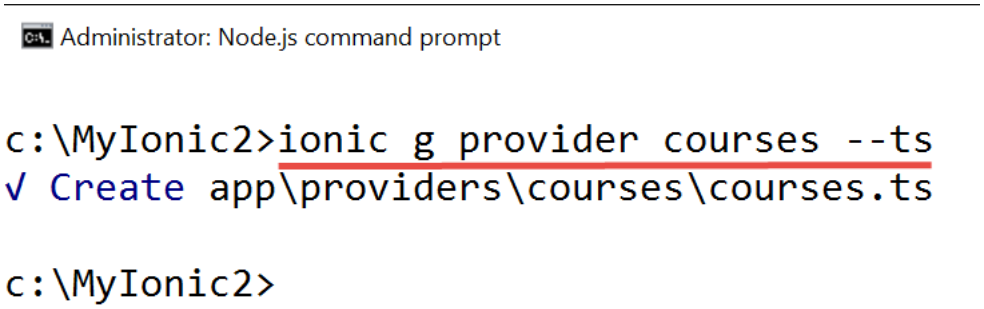
1. เพื่อไม่ให้ซ้ำกันกับหัวข้อที่แล้ว ให้สร้างเพจใหม่ชื่อว่า **my-courses** ใช้คำสั่ง **ionic g page my-courses --ts** แล้วกด Enter และแน่นอนหลังทุกครั้งหลังสร้าง Page อย่าลืมเปิดไฟล์ **app\theme\app.core.scss** แล้วเพิ่มคำสั่ง **@import "../pages/my-courses/my-courses";** ด้วย



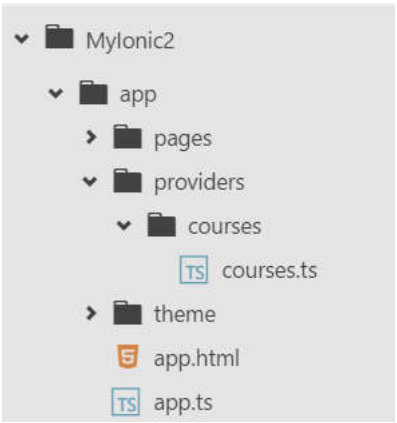
2. สร้าง side menu ตามนี้ (ให้ทดลองเขียนเพิ่มเมนูด้วยตัวเอง)



3. สร้าง provider ใหม่ ด้วยคำสั่ง **ionic g provider courses --ts** แล้วกด Enter



Path ของ providers จะอยู่ที่ app\providers\ชื่อ\ชื่อไฟล์.ts



4. เปิดไฟล์ **app\providers\courses\courses.ts** แล้วใส่ URL http://codingthailand.com/api/get_courses.phpเข้าไปใน method get() ดังนี้

```
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';
import 'rxjs/add/operator/map';

@Injectable()
export class Courses {
```

```
  data: any;

  constructor(private http: Http) {
    this.data = null;
  }

  load() {
    if (this.data) {
      // already loaded data
      return Promise.resolve(this.data);
    }

    // don't have the data yet
    return new Promise(resolve => {
      // We're using Angular Http provider to request the data,
      // then on the response it'll map the JSON data to a parsed JS object.
      // Next we process the data and resolve the promise with the new data.
      this.http.get('http://codingthailand.com/api/get_courses.php')
        .map(res => res.json())
        .subscribe(data => {
          // we've got back the raw data, now generate the core schedule data
          // and save the data for later reference
          this.data = data;
          resolve(this.data);
        });
    });
  }
}
```

Note: Promise ใน JavaScript เป็นการเขียนโปรแกรมแบบ asynchronous ข้อดีคือ เราสามารถตรวจสอบว่างานนั้นๆทำเสร็จหรือไม่ และสามารถดักจับข้อผิดพลาดได้ด้วย

5. เมื่อเราสร้าง providers เรียบร้อย เพจใดๆ ก็ตามที่ต้องการใช้ provider ตัวนี้ สามารถเรียกใช้ได้ตลอดเวลา ในตัวอย่างนี้เราจะใช้เพจ my-courses เพื่อเรียกใช้ provider และแสดงผลข้อมูลกัน ให้เปิดไฟล์ **app\pages\my-courses\my-courses.ts** เขียนโค้ดเพื่อเรียกใช้ provider ดังนี้

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
//import provider ที่ต้องการเข้ามา
import { Courses } from '../../providers/courses/courses';

@Component({
  providers: [Courses], //ระบุชื่อ prodiver เพื่อนำมาใช้ใน class หรือเพจนี้
  templateUrl: 'build/pages/my-courses/my-courses.html',
})
export class MyCoursesPage {
  //ประกาศตัวแปร items เป็น any เราไม่ได้ประกาศเป็น array เพราะ provider คืนค่ามาเป็น Promise
  items: any = null;
  constructor(private nav: NavController, items: Courses) { //ประกาศตัวแปร itmes
      this.items = items.load(); //เรียก method load() จาก prodiver Courses
  }
}
```

6. เปิดไฟล์ **app\pages\my-courses\my-courses.html** เพื่อแสดงผลข้อมูลโดยใช้ List Component

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>รายการคอร์สเรียนฟรี</ion-title>
</ion-navbar>
</ion-header>
```

```
<ion-content  class="courses">
   <ion-list>
     <button ion-item *ngFor="let item of items | async">
       <h2>{{item.c_detail}}</h2>
       <p>{{item.c_title}}</p>
     </button>
   </ion-list>
</ion-content>
```

Note: สังเกตว่าหลังการวนลูป หากเราใช้ provider (Promise) ข้อมูลที่คืนมาจาก method load() จะไม่ได้คืนมาในรูปแบบ array จะคืนมาเป็น Promise Object ดังนั้น ตอนแสดงผลก็ให้ใส่ | **async** เพื่อแปลงข้อมูลด้วย

7. เพียงเท่านี้เราก็สามารถใช้ providers มาช่วยจัดการข้อมูลเรียบร้อยแล้วครับ ลองรันดู

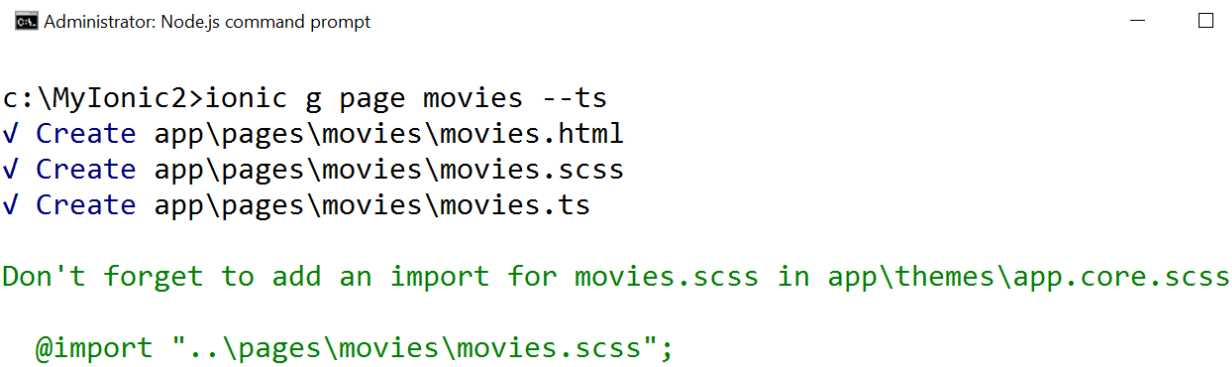
ตอนนี้เรามีความรู้ และสามารถดึงข้อมูลจาก Backend กันได้เรียบร้อย เพื่อทบทวนเราจะมาทำ Workshop กันต่อ โดยจะประยุกต์เพิ่มอีกโดยใช้ Ionic Component อื่นๆ ด้วย เช่น Cards, Loading และจะลองแสดงภาพจาก API ด้วย

ใน Workshop นี้เราจะดึงข้อมูลภาพยนตร์จากเว็บ https://www.cinemalytics.com ซึ่งเค้ามี API ให้เราใช้ด้วย หากอยากใช้ API สามารถสมัครได้ที่ https://developers.cinemalytics.com/ หากไม่สมัครจะลองใช้ auth token เดียวกันกับผมก็ได้ ตาม URL นี้

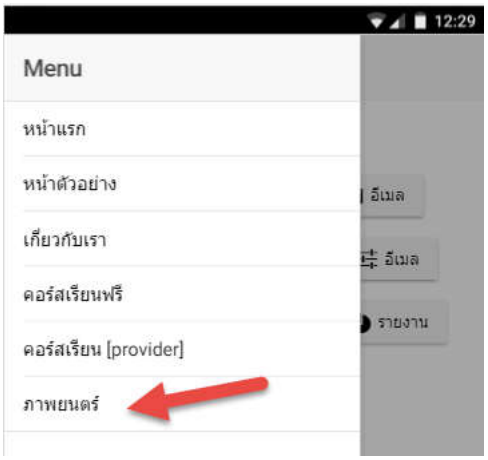
https://api.cinemalytics.com/v1/movie/upcoming?auth_token=6B03E736C94C2E700DB0620966F4D83A

มาลุยกันเลย!

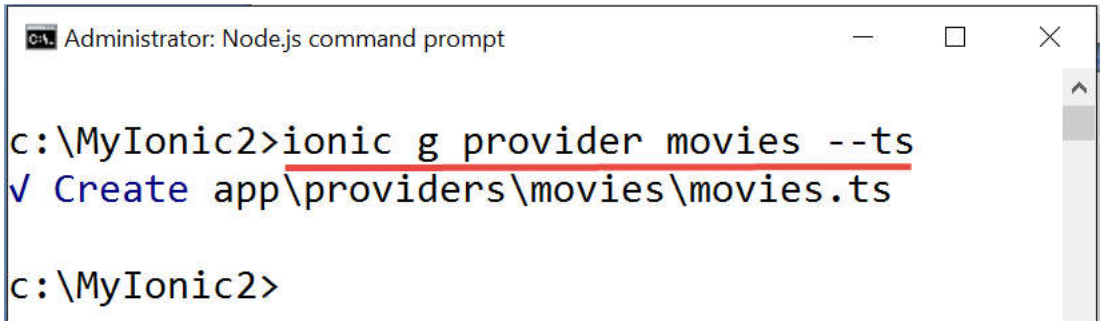
1. ให้สร้างเพจใหม่ชื่อว่า movies ด้วยคำสั่ง **ionic g page movies --ts** แล้วกด Enter เช่นเดิมอย่าลืมหลังทุกครั้งหลังสร้าง Page อย่าลืมเปิดไฟล์ **app\theme\app.core.scss** แล้วเพิ่มคำสั่ง **@import "../pages/movies/movies";** ด้วย



2. สร้าง side menu ตามนี้ (ให้ทดลองเขียนเพิ่มเมนูด้วยตัวเอง)



3. สร้าง provider ใหม่ ด้วยคำสั่ง **ionic g provider movies --ts** แล้วกด Enter



4. เปิดไฟล์ provider **app\providers\movies\movies.ts** แล้วใส่ URL

   https://api.cinemalytics.com/v1/movie/upcoming?auth_token=6B03E736C94C2E700DB0620966F4D83A

   เข้าไปใน method get() ดังนี้

```
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';
import 'rxjs/add/operator/map';
```

```
@Injectable()
export class Movies {
  data: any;

  constructor(private http: Http) {
    this.data = null;
  }

  load() {
    if (this.data) {
      // already loaded data
      return Promise.resolve(this.data);
    }

    // don't have the data yet
    return new Promise(resolve => {
      // We're using Angular Http provider to request the data,
      // then on the response it'll map the JSON data to a parsed JS object.
      // Next we process the data and resolve the promise with the new data.

this.http.get('https://api.cinemalytics.com/v1/movie/upcoming?auth_token=6B03E736C94C2E700DB0620966F4D83A')
        .map(res => res.json())
        .subscribe(data => {
          // we've got back the raw data, now generate the core schedule data
          // and save the data for later reference
          this.data = data;
          resolve(this.data);
        });
    });
  }
}
```

5. เปิดไฟล์ app\pages\movies\movies.ts เพื่อเรียกใช้ provider Movies เขียนโค้ดเพิ่มเติม ดังนี้ ในไฟล์นี้เราจะลองปรับโดยสร้าง method ใหม่ชื่อว่า showData() เพื่อโหลดข้อมูล

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import { Movies } from '../../providers/movies/movies';

@Component({
  providers: [Movies],
  templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
  items: any = null;

  constructor(private nav: NavController, private movie: Movies) {
    this.items = movie.load();
    this.movie = movie;
     this.showData();
  }

  showData() {
     this.items = this.movie.load();
  }
}
```

6. เปิดไฟล์ app\pages\movies\movies.html ในไฟล์นี้จะใช้ Ionic Components ชื่อว่า Cards ในการแสดงผล และทดลองแสดงผลรูปภาพจาก API ด้วย รายละเอียด Cards ดูได้ที่ http://ionicframework.com/docs/v2/components/#cards

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
```

```
  <ion-title>หนังใหม่</ion-title>
</ion-navbar>
</ion-header>

<ion-content  class="movies">

  <ion-card ion-item *ngFor="let item of items | async">
    <img src="{{item.PosterPath}}">
   <ion-card-content>
     <ion-card-title>
       {{ item.Title }}
      </ion-card-title>
     <p>
      {{ item.Description }}
     </p>
     <p>
      {{ item.ReleaseDate }}
     </p>
   </ion-card-content>
  </ion-card>

</ion-content>
```

Note: หากต้องการแสดงรายละเอียดอื่นๆ ก็สามารถปรับได้ตาม API นั้นๆ

7. จากนั้นเราจะลองเพิ่ม Ionic Components ชื่อว่า Loading เพื่อแสดงการรอโหลดข้อมูล รายละเอียดของ Loading ดูได้ที่ http://ionicframework.com/docs/v2/components/#loading

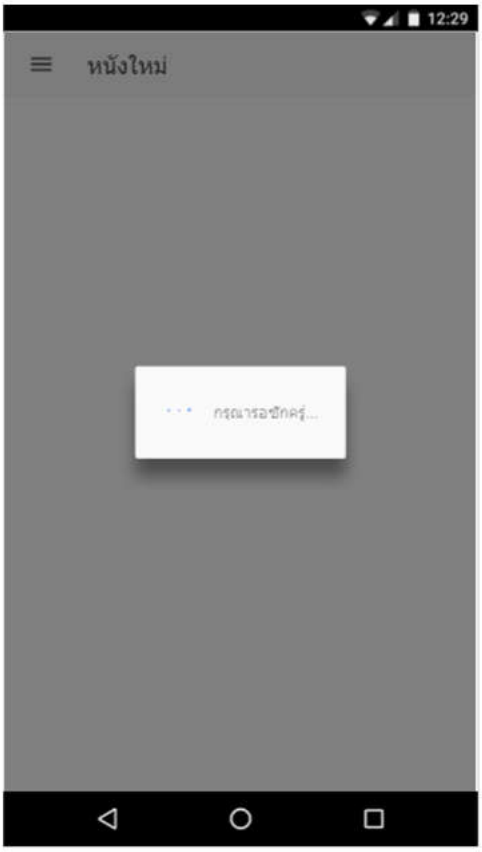
เปิดไฟล์ app\pages\movies\movies.ts แล้วเพิ่มโค้ดส่วนของ Loading ดังนี้

```
import { Component } from '@angular/core';
import { NavController, Loading } from 'ionic-angular';
```

```
import { Movies } from '../../providers/movies/movies';

@Component({
  providers: [Movies],
  templateUrl: 'build/pages/movies/movies.html',
})

export class MoviesPage {
  items: any = null;

  constructor(private nav: NavController, private movie: Movies) {
    this.movie = movie;
     this.showData(); //เรียกใช้ method showData()
  }

  showData() {
    //สร้าง Loading พร้อมกับตั้งค่าต่างๆ
     let loading = Loading.create({
       content: "รอกสักครู่...",
       spinner: 'dots',
       duration: 3000
     });

    //แสดง Loading หากโหลดข้อมูลเรียบร้อยแล้ว ก็ให้หายไป (dismiss)
     this.nav.present(loading).then(() => {
         this.items = this.movie.load();
         loading.dismiss();
     });
  }

}
```

Note: ในตัวอย่างนี้เราสร้าง method showData() เพื่อโหลดข้อมูล และแยกการทำงานออกมา

8. บันทึกไฟล์ แล้วทดสอบรันอีกครั้ง

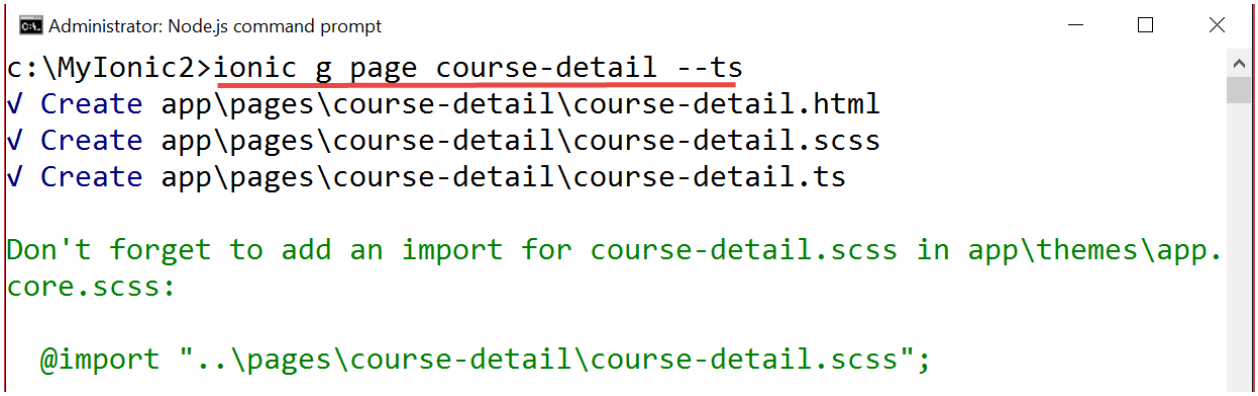


การสร้าง Pages ในรูปแบบ Master Detail

ในหัวข้อที่แล้วเราสร้างได้เพจ ชื่อว่า **courses** ซึ่งโหลดข้อมูลแสดงรายการคอร์สเรียนฟรีจากเว็บ coidngthailand.com เรียบร้อยแล้ว ในหัวข้อนี้เราจะมาทดลองสร้างเพจในรูปแบบ **Master Detail** กันโดยจะให้เพจ courses เป็น Master ที่แสดงรายละเอียดคอร์สเรียนทั้งหมด และจะสร้างเพจใหม่ชื่อว่า **course-detail** เพื่อสำหรับแสดงรายละเอียดคอร์สเมื่อผู้ใช้คลิกเลือกคอร์สเรียนครับ

สำหรับเพจ course-detail นี้เราจะใช้ Backend ตาม URL http://codingthailand.com/api/get_course_detail.php?course_id=1 ตัวเลขด้านหลังสุดของ URL ก็คือ ตัวเลขรหัสคอร์สที่จะต้องรับจากมาจากเพจ courses จะทดลองเปลี่ยนตัวเลขได้ เช่น เปลี่ยนจาก 1 เป็น 2 เป็นต้น มาลุยกัน!

1. ให้สร้างเพจใหม่ชื่อว่า course-detail ด้วยคำสั่ง ionic g page course-detail --ts แล้วกด Enter เช่นเดิมอย่าลืมหลังทุกครั้งหลังสร้าง Page ให้เปิดไฟล์ app\theme\app.core.scss แล้วเพิ่มคำสั่ง @import "../pages/ course-detail/ course-detail";



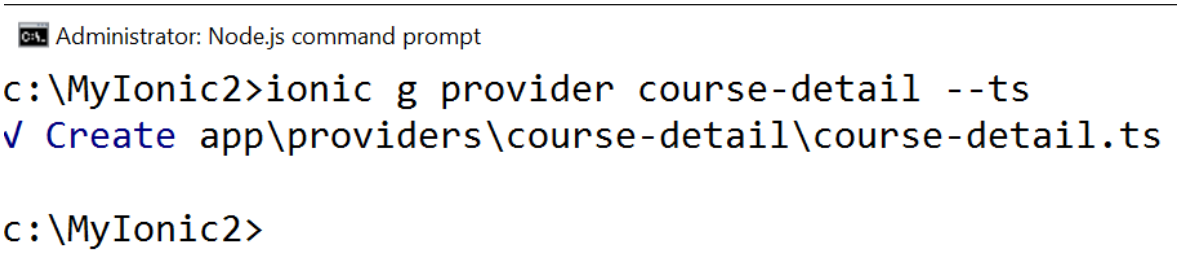
2. ก่อนจะไปที่เพจ course-detail เราจะเขียนในส่วนของเพจ Master ก่อน นั่นคือเพจ courses ให้เปิดไฟล์ app\pages\courses\courses.html เพื่อเขียนโค้ดเหตุการณ์คลิก (click) ให้กับ ion-item เมื่อผู้ใช้คลิกรายการก็ให้เรียกฟังก์ชันที่ชื่อว่า itemTapped พร้อมทั้งส่งตัวแปร item ไปด้วย ดังนี้ (click)="itemTapped(item)"

```
<ion-header>
<ion-navbar>
 <button menuToggle>
  <ion-icon name="menu"></ion-icon>
 </button>
 <ion-title>รายการคอร์สเรียนฟรี</ion-title>
</ion-navbar>
</ion-header>

<ion-content  class="courses">
 <ion-content>
  <ion-list>
   <button ion-item *ngFor="let item of items" (click)="itemTapped(item)">
    <h2>{{item.c_detail}}</h2>
    <p>{{item.c_title}}</p>
   </button>
  </ion-list>
</ion-content>
```

3. เปิดไฟล์ **app\pages\courses\courses.ts** เขียน method itemTapped พร้อมทั้งส่งค่าตัวแปร item ไปยังหน้า **course-detail** การส่งตัวแปร item นั้นจะใช้หลักการการส่งข้อมูลข้ามหน้าหรือระหว่างเพจนั่นเองครับ

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import { HTTP_PROVIDERS, Http } from '@angular/http';
import { CourseDetailPage } from '../course-detail/course-detail';

@Component({
  providers: [HTTP_PROVIDERS],
  templateUrl: 'build/pages/courses/courses.html',
})
export class CoursesPage {
  items: Array<{}>;
  constructor(private nav: NavController, private http: Http) {
    this.http = http;
    this.http.get("http://codingthailand.com/api/get_courses.php").subscribe(
      (response) => {
           this.items = response.json();
        }
       );
  }

  itemTapped(item) {
     this.nav.push(CourseDetailPage, {
       id: item.id,  //ส่งรหัสคอร์สที่เลือกพร้อมกำหนดให้กับ id แล้วส่งไปให้หน้า course-detail
       title: item.c_title //ส่งชื่อคอร์สไปที่เลือกพร้อมกำหนดให้กับ title แล้วส่งไปให้หน้า course-detail
     });
  }

}
```

4. สร้าง provider ใหม่เพื่อให้กับเพจ course-detail ด้วยคำสั่ง **ionic g provider course-detail --ts** แล้วกด Enter



5. เปิดไฟล์ provider **app\providers\course-detail\course-detail** แล้วใส่ URL

https://api.cinemalytics.com/v1/movie/upcoming?auth_token=6B03E736C94C2E700DB0620966F4D83A

เข้าไปใน method get() ดังนี้

```
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';
import 'rxjs/add/operator/map';

@Injectable()
export class CourseDetail {
  data: any;
  url: string;

  constructor(private http: Http) {
    this.data = null;
  }

  load(id:number) { //สร้างพารามิเตอร์ ตัวแปร id เพื่อส่งรหัสคอร์สเข้ามาดึงข้อมูลของคอร์สนั้นๆ
    if (this.data) {
      // already loaded data
      return Promise.resolve(this.data);
    }

    // don't have the data yet
    return new Promise(resolve => {
      // We're using Angular Http provider to request the data,
```

```
    // then on the response it'll map the JSON data to a parsed JS object.
    // Next we process the data and resolve the promise with the new data.
    //สร้างตัวแปร url ในรูปแบบของ template string ใช้เครื่องหมาย backtick (`) พร้อมส่งตัวแปร id เข้าไป
    this.url = `http://codingthailand.com/api/get_course_detail.php?course_id=${id}`;
    this.http.get(this.url)
      .map(res => res.json())
      .subscribe(data => {
        // we've got back the raw data, now generate the core schedule data
        // and save the data for later reference
        this.data = data;
        resolve(this.data);
      });
  });
 }
}
```

6. เปิดไฟล์ app\pages\course-detail\course-detail.ts เพื่อรับค่าที่ส่งมาจากเพจ courses พร้อมทั้งโหลดหัวข้อการสอนในคอร์สต่างๆ

```
import { Component } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';
import { CourseDetail } from '../../providers/course-detail/course-detail';

@Component({
 providers: [CourseDetail],
 templateUrl: 'build/pages/course-detail/course-detail.html',
})
export class CourseDetailPage {
 id: number;
 title: string;
 items: any = null;
```

```
  constructor(private nav: NavController, private navParams: NavParams, items: CourseDetail) {
    this.nav = nav;
    this.id = navParams.get('id'); //id ส่งมาจากเพจ courses.ts
    this.title = navParams.get('title'); //title ส่งมาจากเพจ courses.ts
    this.items = items.load(this.id); //ส่ง argument ตัวแปร id เข้าไปเพื่อดึงเฉพาะคอร์สที่รับมา
  }
}
```

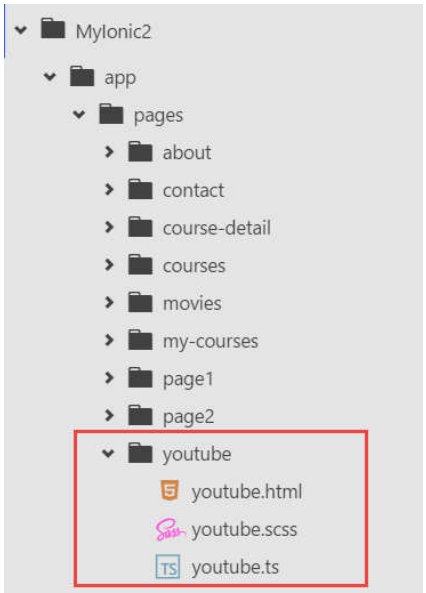
7. เปิดไฟล์ app\pages\course-detail\course-detail.html เพื่อเขียนโค้ดแสดงผลข้อมูล

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>คอร์ส {{ title }}</ion-title>
</ion-navbar>
</ion-header>

<ion-content>
  <ion-list>
    <button ion-item *ngFor="let item of items | async" (click)="itemTapped(item)">
      <h2>{{item.ch_title}}</h2>
    </button>
  </ion-list>
</ion-content>
```

8. บันทึกไฟล์ แล้วทดสอบ App โดยลองกดเลือกที่คอร์สเรียน เมื่อกดแล้วจะเห็นหัวข้อเรียนในคอร์สนั้นๆ ครับ

ถึงตรงนี้แล้วเราจะมาลองสร้างอีก 1 เพจเพื่อให้แสดงวิดีโอจาก Youtube กัน คือเมื่อคลิกที่หัวข้อเรียนแล้วให้เปิดหน้าเพจ youtube ขึ้นมาแสดงครับ แต่! ในหัวข้อนี้ผมจะให้ทำแบบฝึกหัดในการสร้างเพจเองนะครับ จะให้ขั้นตอนไว้เฉยๆ มาเริ่มกันเลย

1. สร้างเพจชื่อว่า youtube (ให้สร้างเองนะครับ)



2. เปิดไฟล์ app\course-detail\course-detail.ts เพื่อส่งค่า URL ของ youtube และรายละเอียดต่างๆไปให้เพจ youtube

```
import { Component } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';
import { CourseDetail } from '../../providers/course-detail/course-detail';
import { YoutubePage } from '../youtube/youtube';

@Component({
  providers: [CourseDetail],
  templateUrl: 'build/pages/course-detail/course-detail.html',
})
export class CourseDetailPage {
  id: number;
  title: string;
  items: any = null;

  constructor(private nav: NavController, private navParams: NavParams, items: CourseDetail) {
    this.nav = nav;
    this.id = navParams.get('id'); //id ส่งมาจากเพจ courses.ts
    this.title = navParams.get('title'); //title ส่งมาจากเพจ courses.ts
    this.items = items.load(this.id); //ส่ง argument ตัวแปร id เข้าไปเพื่อดึงเฉพาะคอร์สที่รับมา
  }

  itemTapped(item) {
    this.nav.push(YoutubePage, {
        youtubeUrl: item.ch_url, //ส่ง url ของวิดีโอ youtube ไปที่เพจ youtube
        chTitle: item.ch_title, //ส่ง หัวข้อสอน ไปที่เพจ youtube
        chView: item.ch_view, //ส่ง ยอดผู้เข้าชม ไปที่เพจ youtube
        chDateAdd: item.ch_dateadd //ส่ง วันที่เพิ่มวิดีโอ ไปที่เพจ youtube
    });
  }
```

```
}
```

3. เปิดไฟล์ app\youtube\youtube.ts เพื่อทำการรับค่าที่ส่งมาจากเพจ course-detail

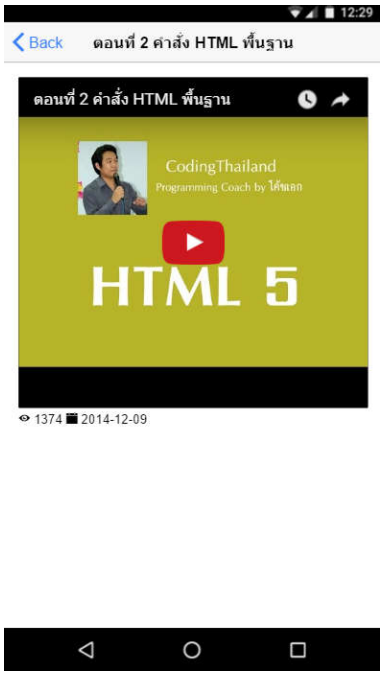
```
import { Component } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';
import {DomSanitizationService, SafeResourceUrl} from '@angular/platform-browser';

@Component({
  templateUrl: 'build/pages/youtube/youtube.html',
})
export class YoutubePage {
  youtubeUrl: string;
  chTitle: string;
  chView: number;
  chDateAdd: string;
  url: SafeResourceUrl;
  newUrl: string;

  constructor(private nav: NavController, private params: NavParams, sanitizer: DomSanitizationService) {
    //รับค่าต่างๆมาจากเพจ course-detail (course-detail.ts)
    this.newUrl = "https://www.youtube.com/embed/"+params.get('youtubeUrl');
    this.url = sanitizer.bypassSecurityTrustResourceUrl(this.newUrl);
    this.chTitle = params.get('chTitle');
    this.chView = params.get('chView');
    this.chDateAdd = params.get('chDateAdd');
  }
}
```

Note: ตั้งแต่ Angular rc.2 เป็นต้นมา Angular 2 จะเข้มงวดเรื่องความปลอดภัยมากขึ้น ในเพจนี้เราใช้ <iframe> ซึ่งถูกมองว่าไม่ปลอดภัย เราจึงต้องอนุญาต URL ด้วยโดยใช้ method bypassSecurityTrustResourceUrl() ครับ

4. เปิดไฟล์ app\youtube\youtube.html เพื่อแสดงวิดีโอ และรายละเอียดต่างๆ

```
<ion-header>
<ion-navbar>
 <button menuToggle>
  <ion-icon name="menu"></ion-icon>
 </button>
 <ion-title>{{ chTitle }}</ion-title>
</ion-navbar>
</ion-header>

<ion-content>
 <iframe width="100%" height="360" [src]="url" frameborder="0" allowfullscreen></iframe>
  <ion-icon name="eye"></ion-icon> {{chView}}
  <ion-icon name="calendar"></ion-icon> {{chDateAdd}}
</ion-content>
```

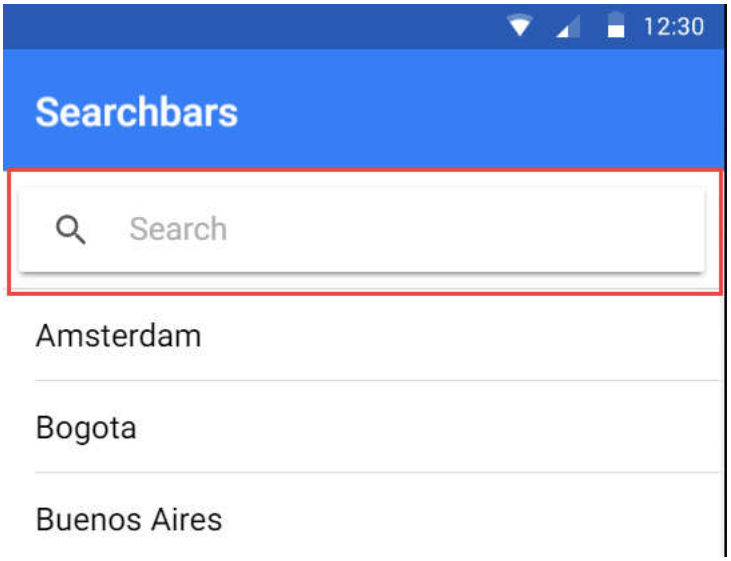
5. บันทึกไฟล์ ทดลองรัน App



Note: เมื่อเรากดเล่น Youtube บน localhost จะไม่สามารถเล่นได้ แต่ถ้าเอาขึ้น store จะไม่มีปัญหาครับ (ปัญหาเรื่อง cross-origin resource)

การใช้ Component Searchbar สำหรับค้นหาข้อมูล

หลังจากที่เราสร้าง Lists Component กันมาพอสมควร หากเราต้องการค้นหาข้อมูล หรือกรองข้อมูล แนะนำให้ใช้ Searchbar ครับ



Note: รายละเอียด Searchbar ดูได้ที่ http://ionicframework.com/docs/v2/components/#searchbar

เมื่อผู้ใช้ค้นข้อมูล เราจะต้องใช้ฟังก์ชันของ JavaScript เข้ามาช่วยในการค้นหา เช่น indexOf, filter, includes เป็นต้น การใช้งานฟังก์ชันพวกนี้อยู่ในพื้นฐานของ JavaScript อยู่แล้ว ในตัวอย่างนี้เราจะลองทำ Searchbar เพื่อค้นหาข้อมูลคอร์สเรียนกัน เริ่มเลย!

1. เปิดไฟล์ **app\pages\courses\courses.html** เพื่อแทรก Searchbar Component เข้าไป

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>รายการคอร์สเรียนฟรี</ion-title>
</ion-navbar>
</ion-header>

<ion-content class="courses">
<ion-searchbar [(ngModel)]="searchQuery" [cancelButtonText]="cancelText" [placeholder]="placeholder"
(input)="searchItems($event)"></ion-searchbar>
```

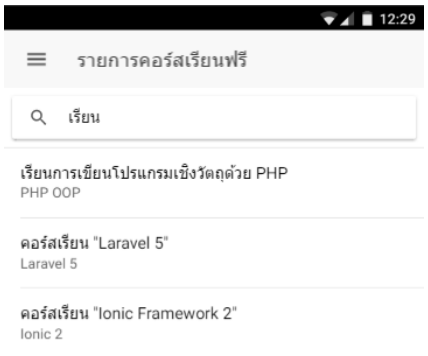
```
    <ion-list>
      <button ion-item *ngFor="let item of items" (click)="itemTapped(item)">
        <h2>{{item.c_detail}}</h2>
        <p>{{item.c_title}}</p>
      </button>
    </ion-list>
  </ion-content>
```

2. เปิดไฟล์ **app\pages\courses\courses.ts** สร้าง method **searchItems()** เพื่อช่วยค้นหาข้อมูล ดังนี้

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import { HTTP_PROVIDERS, Http } from '@angular/http';
import { CourseDetailPage } from '../course-detail/course-detail';

@Component({
  providers: [HTTP_PROVIDERS],
  templateUrl: 'build/pages/courses/courses.html',
})
export class CoursesPage {
  items: any = null;
  cancelText: string;
  placeholder: string;

  constructor(private nav: NavController, private http: Http) {
    this.http = http;
    this.showData();
    this.cancelText = 'ยกเลิก'; //กำหนด label ให้กับปุ่มยกเลิก
    this.placeholder = 'ค้นหาคอร์สเรียน';
  }

 itemTapped(item) {
```

```
    this.nav.push(CourseDetailPage, {
      id: item.id,  //ส่งรหัสคอร์สที่เลือกพร้อมกำหนดให้กับ id แล้วส่งไปให้หน้า course-detail
      title: item.c_title //ส่งชื่อคอร์สไปที่เลือกพร้อมกำหนดให้กับ title แล้วส่งไปให้หน้า course-detail
    });
  }

  showData() {
    this.http.get("http://codingthailand.com/api/get_courses.php").subscribe(
      (response) => {
          this.items = response.json();
        }
       );
  }

  searchItems(event) {
     var val = event.target.value; //ดึงค่าจาก event ที่ส่งมาจาก .html
     if (val && val.trim() != '') {
         this.items = this.items.filter((item) => item.c_detail.toLowerCase().includes(val.toLowerCase()));
     } else {
         this.items = this.showData();
     }
  }
}
```

3. ทดสอบรัน App แล้วลองพิมพ์ค้นหา



บทที่ 8 การสร้างฟอร์ม (Form) และ Dialogs

ในการสร้าง Mobile App มีหลายครั้งที่เราต้องให้ผู้ใช้กรอกข้อมูลผ่านฟอร์ม เช่น ฟอร์มสมัครสมาชิก ฟอร์มล็อกอิน ฟอร์มเพิ่มข้อมูล เป็นต้น ใน Ionic 2 นั้น การสร้างฟอร์มต่างๆ จะใช้ความสามารถของ Angular 2 หากอยากศึกษาเพิ่มเติมให้ค้นเรื่อง Angular 2 Form ได้ครับ

ลองสร้างฟอร์มสมัครสมาชิก

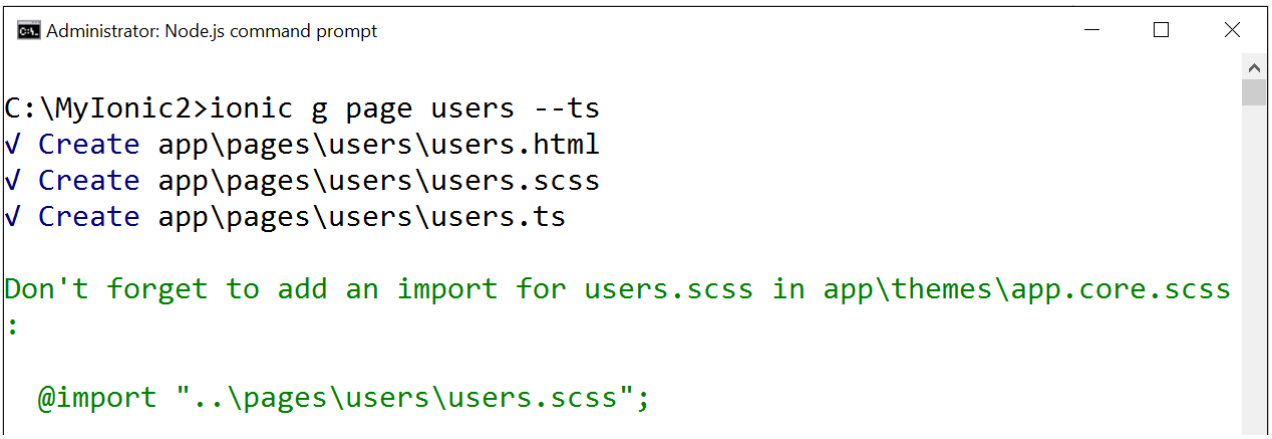
ในหัวข้อนี้เราจะมาสร้างฟอร์มสมัครสมาชิก เมื่อผู้ใช้กรอกข้อมูลแล้วจะไปบันทึกข้อมูลที่ Backend และจะให้แสดง Alert เมื่อบันทึกข้อมูลเรียบร้อยแล้ว เราจะใช้ URL http://codingthailand.com/api/insert_user.php เพื่อใช้สำหรับเพิ่มข้อมูลลงในตารางผู้ใช้ และจะใช้ URL http://codingthailand.com/api/get_user.php สำหรับดูข้อมูลผู้ใช้ ไฟล์ insert_user.php มีโค้ด ดังนี้

```
<?php

        $link = mysqli_connect('localhost', 'root', '', 'mydb');
        mysqli_set_charset($link, 'utf8');

        $fullname = $_POST['fullname'];
        $username = $_POST['username'];
        $password = $_POST['password'];

        //insert data
        $sql = "INSERT INTO `user` (`id`, `fullname`, `username`, `password`) VALUES (NULL,'$fullname', '$username',
'$password')";
        $result = mysqli_query($link, $sql);

        if ($result) {
    echo json_encode(array('status' => 'success','message' => 'บันทึกข้อมูลเรียบร้อยแล้ว'));
        } else {
    echo json_encode(array('status' => 'errors','message' => 'เกิดข้อผิดพลาด'));
        }
```

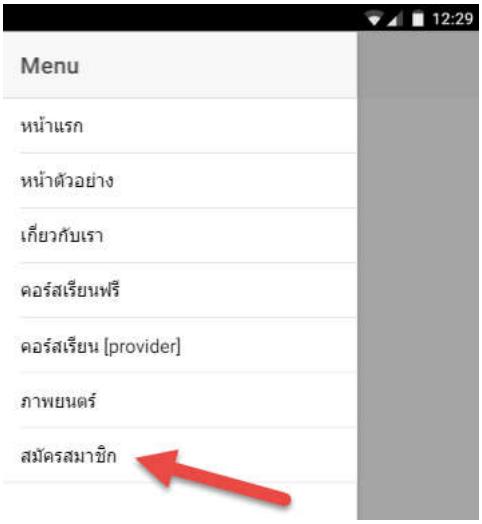
Note: ไฟล์ insert_user.php ใช้เป็นตัวอย่างเท่านั้น รหัสผ่านควรมีการเข้ารหัสเสมอ

ขั้นตอนการสร้างฟอร์ม มีดังนี้

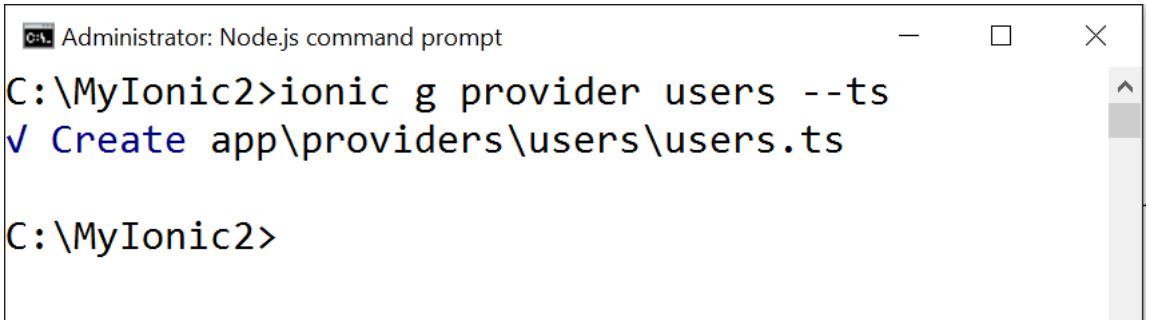
1. ให้สร้างเพจใหม่ชื่อว่า users ด้วยคำสั่ง ionic g page users --ts แล้วกด Enter เช่นเดิมอย่าลืมหลังทุกครั้งหลังสร้าง Page อย่าลืมเปิดไฟล์ app\theme\app.core.scss แล้วเพิ่มคำสั่ง @import "../pages/users/users"; ด้วย



2. สร้าง side menu ตามนี้ (ให้ทดลองเขียนเพิ่มเมนูด้วยตัวเอง)



3. สร้าง provider ใหม่ ด้วยคำสั่ง ionic g provider users --ts แล้วกด Enter



4. เปิดไฟล์ app\pages\users\users.html เพื่อออกแบบฟอร์ม มีโค้ด ดังนี้

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>สมัครสมาชิก</ion-title>
</ion-navbar>
</ion-header>

<ion-content padding>

  <form [ngFormModel]="memberForm" (submit)="addMember($event)">
    <ion-list>

      <ion-item>
        <ion-label floating>Full Name</ion-label>
        <ion-input ngControl="fullname" type="text"></ion-input>
      </ion-item>

      <ion-item>
        <ion-label floating>Username</ion-label>
        <ion-input ngControl="username" type="text"></ion-input>
      </ion-item>

      <ion-item>
        <ion-label floating>Password</ion-label>
        <ion-input ngControl="password" type="password"></ion-input>
      </ion-item>
      <br>
      <button type="submit" [disabled]="!memberForm.valid">สมัครสมาชิก</button>
```

```
    </ion-list>
  </form>

</ion-content>
```

5. เปิดไฟล์ app\pages\users\users.ts เพื่อเขียนโค้ด บันทึกข้อมูล และตรวจสอบความถูกต้องของข้อมูล ดังนี้

```
import { Component } from '@angular/core';
import { NavController, Alert } from 'ionic-angular';
import { Users } from '../../providers/users/users';

//import ตัวช่วยสร้างฟอร์ม การตรวจสอบความถูกต้องของข้อมูล
import {
    FormBuilder,
    Validators,
    ControlGroup,
    FORM_DIRECTIVES,
} from '@angular/common';

@Component({
  providers: [Users],
  directives: [FORM_DIRECTIVES],
  templateUrl: 'build/pages/users/users.html',
})
export class UsersPage {
  memberForm: ControlGroup;

  constructor(private nav: NavController, private fb: FormBuilder, private user: Users) {
    this.nav = nav;
    this.user = user;

    this.memberForm = fb.group({
```

```
      username: ["", Validators.required], //ผู้ใช้จะต้องกรอก
      fullname: ["", Validators.required], //ผู้ใช้จะต้องกรอก
      //password: ["", Validators.required]
      //ด้านล่างเป็นการตรวจสอบความถูกต้องของข้อมูลทีละหลายตัว
      password: ["", Validators.compose([Validators.required,Validators.minLength(4)])]
    });
  }

//method เพิ่มสมาชิกผู้ใช้
  addMember(event) {
    //console.log(this.memberForm.value);
    //console.log(JSON.stringify(this.memberForm.value));
    //ดึงค่ามาจากฟอร์มแต่ละตัว
     let fullname = this.memberForm.controls['fullname'].value;
     let username = this.memberForm.controls['username'].value;
     let password = this.memberForm.controls['password'].value;
    //console.log(fullname);
    //console.log(username);
    //console.log(password);

    //กำหนดตัวแปร body เพื่อส่งให้ method save() ของ provider เพื่อบันทึกข้อมูล
     let body = "fullname="+fullname+"&username="+username+"&password="+password;
    this.user.save(body).then((response)=>{
      //console.log(response.message); //message prop. from server
      //สั่งให้ Alert ทำงานเมื่อบันทึกข้อมูลเรียบร้อย
      let alert = Alert.create({
        title: 'ผลการทำงาน',
        subTitle: response.message, //message เป็นค่าที่ส่งกลับมาจาก Backend
        buttons: ['ตกลง']
      });
      this.nav.present(alert);
```

```
    }).catch((err)=>{
        console.log(err);
    });

    event.preventDefault();
  }

}
```

6. เปิดไฟล์ provider **app\providers\users\users.ts** เพื่อกำหนด URL ในการดึงข้อมูลผู้ใช้ และสร้าง method save() เพื่อเพิ่มข้อมูลผู้ใช้

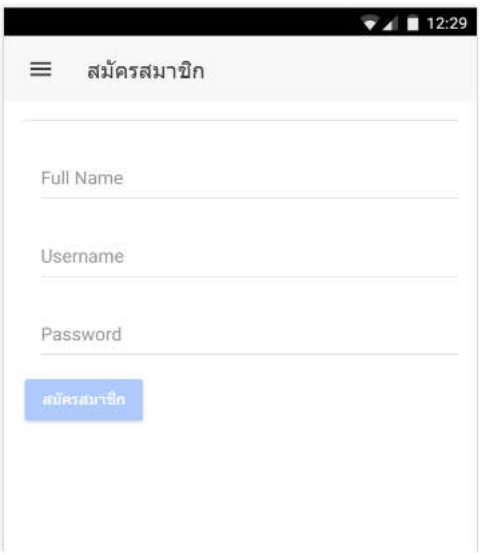
```
import {Injectable} from '@angular/core';
import {Http, Headers} from '@angular/http';
import 'rxjs/add/operator/map';

@Injectable()
export class Users {
  data: any = null;

  constructor(public http: Http) {}

  load() {
    if (this.data) {
      // already loaded data
      return Promise.resolve(this.data);
    }

    // don't have the data yet
    return new Promise(resolve => {
      // We're using Angular Http provider to request the data,
      // then on the response it'll map the JSON data to a parsed JS object.
```

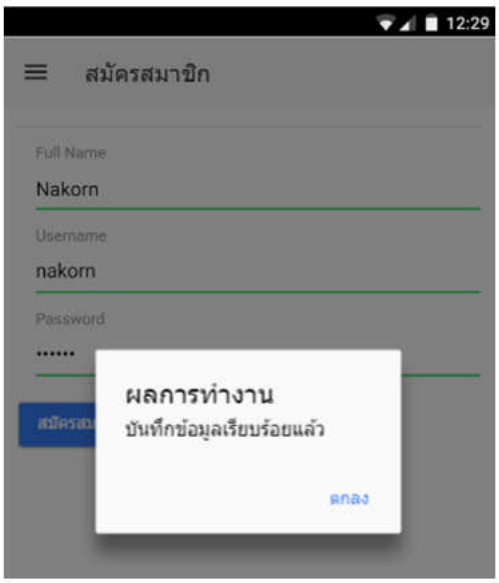
```
    // Next we process the data and resolve the promise with the new data.
    this.http.get('http://www.codingthailand.com/api/get_user.php')
      .map(res => res.json())
      .subscribe(data => {
        // we've got back the raw data, now generate the core schedule data
        // and save the data for later reference
        this.data = data;
        resolve(this.data);
      });
  });
 }

//method สำหรับส่งข้อมูลจากฟอร์มไปบันทึกที่ Backend
  save(body:string) {
   let headers = new Headers();
   headers.append('Content-Type','application/x-www-form-urlencoded; charset=UTF-8');
   if (this.data) {
     // already loaded data
     return Promise.resolve(this.data);
   }

   // don't have the data yet
   return new Promise(resolve => {
     // We're using Angular Http provider to request the data,
     // then on the response it'll map the JSON data to a parsed JS object.
     // Next we process the data and resolve the promise with the new data.
      this.http.post('http://www.codingthailand.com/api/insert_user.php',body,{headers: headers})
       .map(res => res.json())
       .subscribe(data => {
         // we've got back the raw data, now generate the core schedule data
         // and save the data for later reference
         this.data = data;
```

```
            resolve(this.data);
          });
        });
      }

    }
```

Note: หากเป็นการเพิ่มข้อมูลจากใช้ post() แทน get()

7. ทดลองรัน App เสร็จแล้วกรอกข้อมูลแล้วคลิกปุ่มสมัครสมาชิกเพื่อทดสอบ





8. หากต้องการดูข้อมูลที่กรอกไปแล้วสามารถเข้าดูได้ตาม URL http://codingthailand.com/api/get_user.php

บทที่ 9 การใช้ SQLite สำหรับเก็บข้อมูลแบบ Local Storage

การเก็บข้อมูลใน Ionic 2 (Storage) มี 2 แบบด้วยกัน ได้แก่

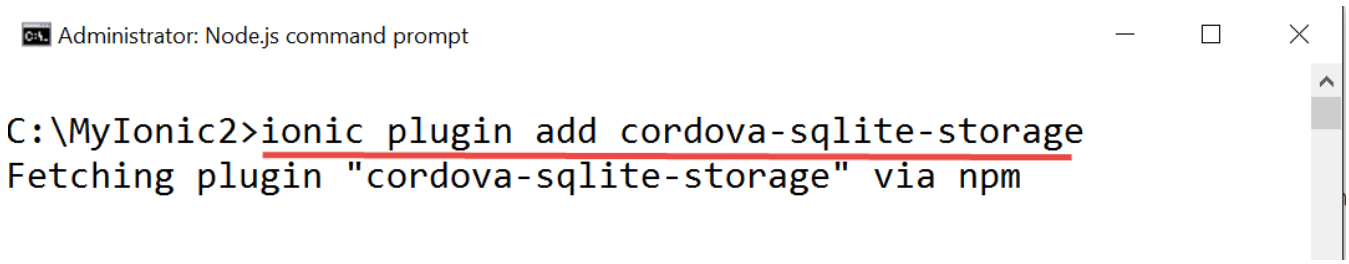
1. LocalStorage
2. SqlStorage

หากเราต้องการเก็บข้อมูลไว้ในเครื่องผู้ใช้ หรือใช้เป็นฐานข้อมูลเก็บข้อมูลบางอย่าง Ionic 2 เตรียมมาให้เราใช้เรียบร้อยแล้ว ลำดับแรกเราสามารถใช้ Local Storage (HTML5) ได้ แต่มันจะจำกัดขนาดแค่ 5MB เท่านั้น ตัวนี้จะเหมาะกับข้อมูลที่ไม่ใหญ่มากนัก

ส่วน SqlStorage จะใช้ฐานข้อมูล SQLite มาช่วยเก็บข้อมูลซึ่งแน่นอนเราสามารถใช้ได้ไม่จำกัดขนาด และยังสามารถใช้ภาษา SQL ไม่ว่าจะเป็น SELECT, INSERT, UPDATE, DELETE มาช่วยในการจัดการข้อมูลได้อีกด้วย ดังนั้นจึงแนะนำวิธีนี้

การใช้งาน SqlStorage

ก่อนจะใช้งาน SqlStorage เราจะต้องติดตั้ง plugin ก่อน โดยใช้คำสั่ง ดังนี้ **ionic plugin add cordova-sqlite-storage** จากนั้นกด Enter เพื่อติดตั้ง



ขั้นตอนการใช้งาน มีดังต่อไปนี้

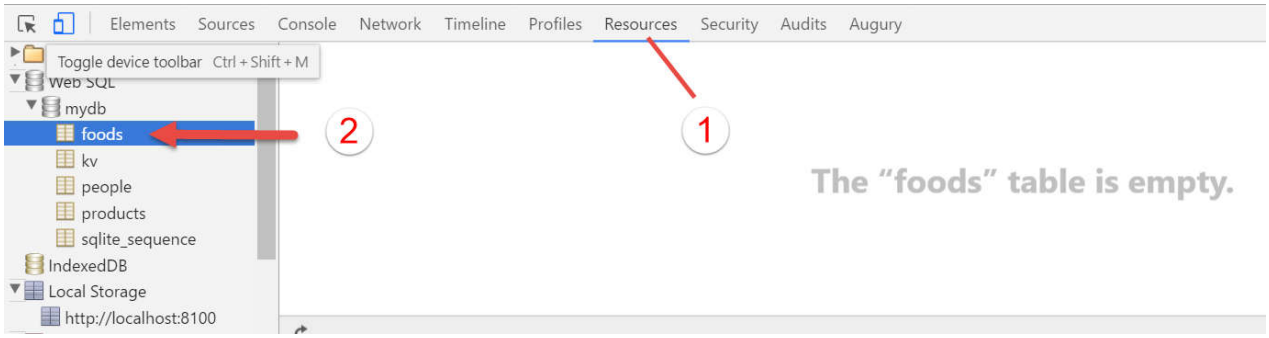
1. สร้างเพจชื่อว่า foods ด้วยคำสั่ง **ionic g page foods --ts** แล้วกด Enter อย่าลืมหลังทุกครั้งหลังสร้าง Page ให้เปิดไฟล์ **app\theme\app.core.scss** แล้วเพิ่มคำสั่ง @import "../pages/foods/foods"; ด้วย
2. ก่อนที่จะใช้งานเราจะต้องสร้างฐานข้อมูล และตารางต่างๆก่อน โดยปกติแล้วเราจะสร้างตอน App เริ่มทำงานนั่นเอง เปิดไฟล์ **app\app.ts** เพิ่มการ import แล้วเขียนสร้างฐานข้อมูล ตารางใหม่ ดังนี้

```
import { Component, ViewChild } from '@angular/core';
import { App, ionicBootstrap, Platform, Nav, Storage, SqlStorage } from 'ionic-angular';
import { StatusBar } from 'ionic-native';

import { Page1 } from './pages/page1/page1';
```

```
import { Page2 } from './pages/page2/page2';
import { AboutPage } from './pages/about/about';
import { CoursesPage } from './pages/courses/courses';
import { MyCoursesPage } from './pages/my-courses/my-courses';
import { MoviesPage } from './pages/movies/movies';
import { UsersPage } from './pages/users/users';
import { FoodsPage } from './pages/foods/foods';

@Component({
  templateUrl: 'build/app.html'
})
class MyApp {
  @ViewChild(Nav) nav: Nav;

  rootPage: any = Page1;

  pages: Array<{title: string, component: any}>

  sql: Storage;

  constructor(private platform: Platform) {
    this.initializeApp();
    //สร้างฐานข้อมูลใหม่ ชื่อว่า mydb
    this.sql = new Storage(SqlStorage,{name: 'mydb'});
    //สร้างตาราง ชื่อว่า foods ประกอบด้วย id (auto) , name, price
    this.sql.query('CREATE TABLE IF NOT EXISTS foods (id INTEGER PRIMARY KEY AUTOINCREMENT, name
TEXT, price INTEGER)');

    // used for an example of ngFor and navigation
    this.pages = [
      { title: 'หน้าแรก', component: Page1 },
      { title: 'หน้าตัวอย่าง', component: Page2 },
      { title: 'เกี่ยวกับเรา', component: AboutPage },
```

```
      { title: 'คอร์สเรียนฟรี', component: CoursesPage },
      { title: 'คอร์สเรียน [provider]', component: MyCoursesPage },
      { title: 'ภาพยนตร์', component: MoviesPage },
      { title: 'สมัครสมาชิก', component: UsersPage },
      { title: 'รายการอาหาร', component: FoodsPage }
    ];

  }

  initializeApp() {
    this.platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
      StatusBar.styleDefault();
    });
  }

  openPage(page) {
    // Reset the content nav to have just this page
    // we wouldn't want the back button to show in this scenario
    this.nav.setRoot(page.component);
  }
}

ionicBootstrap(MyApp);
```

เสร็จแล้วลองทดสอบรัน App จะเห็นว่ามีการสร้างฐานข้อมูลใหม่ และตารางเรียบร้อยแล้ว (ลอง inspect ดู)



3. เปิดไฟล์ app\pages\foods\foods.ts เพื่อเขียนสำหรับเพิ่ม และแสดงข้อมูล

```
import { Component } from '@angular/core';
import { NavController, Storage, SqlStorage } from 'ionic-angular';

@Component({
  templateUrl: 'build/pages/foods/foods.html',
})
export class FoodsPage {
 sql: Storage;
 items: Array<{}>;

constructor(private nav: NavController) {
  this.sql = new Storage(SqlStorage,{name: 'mydb'});//เลือกฐานข้อมูล
  //เพิ่มข้อมูล 3 รายการ
  this.sql.query("INSERT INTO foods (name,price) VALUES ('ข้าวผัด',100)");
  this.sql.query("INSERT INTO foods (name,price) VALUES ('ข้าวกระเพราหมู',300)");
  this.sql.query("INSERT INTO foods (name,price) VALUES ('ยำรวมมิตร',200)");
  //เพิ่มเสร็จให้แสดงข้อมูลทั้งหมด
  this.showData();
}

//method สำหรับแสดงผล ใช้คำสั่ง SQL (SELECT)
showData() {
  this.sql.query("SELECT * FROM foods").then((data)=>{
     //console.log(data.res.rows.item(0).name);
```

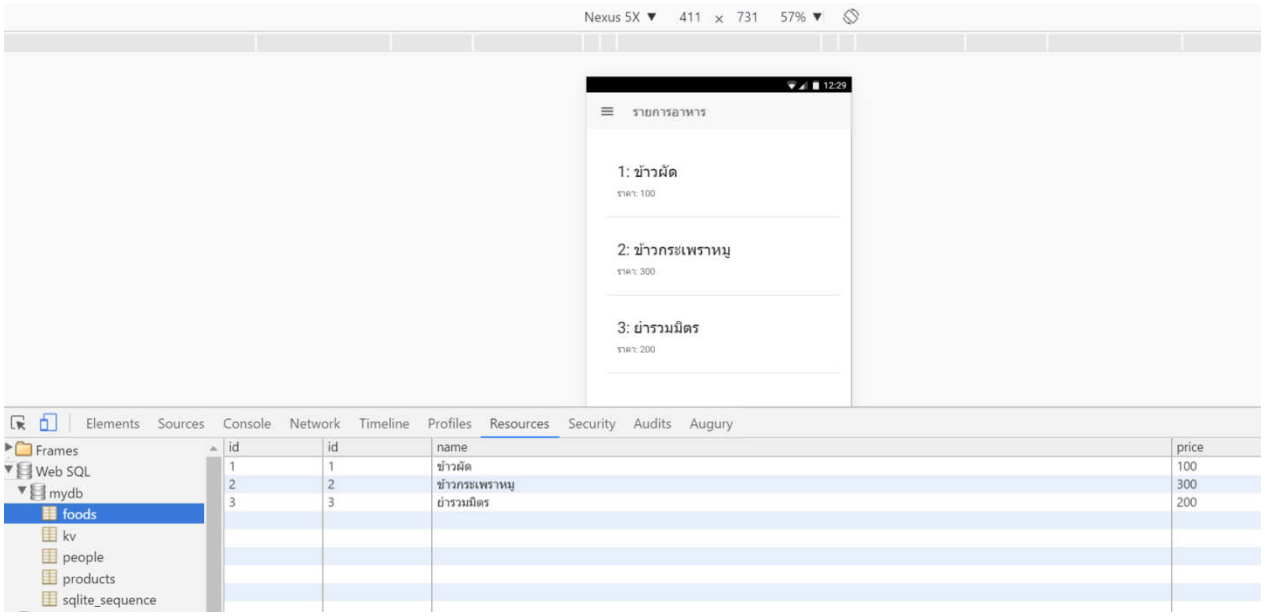
```
      //console.log(data.res.rows.item(0).price);
      if (data.res.rows.length > 0) {
        this.items = [];
        //วนลูปจากแถวในตาราง และ push เข้า array
        for (let i = 0; i < data.res.rows.length; i++) {
            this.items.push({
            id: data.res.rows.item(i).id,
            name: data.res.rows.item(i).name,
            price: data.res.rows.item(i).price
          });
        }
        console.log(this.items);
      }
  });
}

}
```

4. เปิดไฟล์ app\pages\foods\foods.html เขียนโค้ดเพื่อแสดงผลข้อมูลจากตารางในรูปแบบของ Cards

```
<ion-header>
<ion-navbar>
  <button menuToggle>
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>รายการอาหาร</ion-title>
</ion-navbar>
</ion-header>

<ion-content padding>
  <ion-card ion-item *ngFor="let item of items">
    <ion-card-content>
```

```
      <ion-card-title>
        {{ item.id }}: {{item.name}}
      </ion-card-title>
      <p>
        ราคา: {{item.price}}
      </p>
    </ion-card-content>
  </ion-card>

</ion-content>
```

5. ทดสอบ App โดยกดเลือกที่เมนูรายการอาหาร จะเห็นว่ามีเมนูรายการอาหารถูกเพิ่มข้อมูลเรียบร้อย



Note: รายละเอียดเพิ่มเติม http://ionicframework.com/docs/v2/api/platform/storage/SqlStorage/

บทที่ 10 การตกแต่ง App ด้วย Sass และ จัดรูปแบบการแสดงข้อมูลด้วย Pipes

การตกแต่งความสวยงามให้กับ App ก็คือการทำ Theme นั่นเอง ปกติแล้วถ้าจะให้ดีแนะนำศึกษาเพิ่มเติมเรื่อง Sass ครับ หากเราไม่เขียน Sass ในเพจต่างก็สามารถใช้ CSS ปกติได้ครับ ไม่มีปัญหาแต่อย่างใด

การทำ Theming ให้กับ App

ถ้าต้องการเปลี่ยนสีต่างๆของ Theme ให้เปิดไฟล์ app/theme/app.variables.scss แล้วแก้ไขค่าสีได้เลยครับ

```
$colors: (
  primary:    #387ef5,
  secondary:  #32db64,
  danger:     #f53d3d,
  light:      #f4f4f4,
  dark:       #222,
  favorite:   #69BB7B
);
```

เราสามารถตั้งชื่อตัวแปรเพิ่มได้ และค่าสีเพิ่มได้ เช่น

```
$colors: (
  // ...
  twitter:    #55acee
)
```

เวลาเรียกใช้ก็ระบุชื่อตัวแปรให้กับ Components นั้นได้เลย เช่น

```
<button twitter>I'm a button</button>
```

Utility Attributes

Ionic 2 ได้เตรียมตัวช่วยเพื่อให้เราเขียน Sass/CSS ได้ง่ายขึ้น โดยที่เราไม่ต้องเขียนเอง ก่อนเขียนเองแนะนำให้ดูหัวข้อนี้ก่อนครับ เราสามารถใส่ Attributes เหล่านี้ไปที่ Components ที่เราต้องการได้เลย ประกอบด้วย

1. **การจัดรูปแบบข้อความ ที่ใช้บ่อย**

text-left	จัดข้อความชิดซ้าย
text-center	จัดข้อความกึ่งกลาง
text-right	จัดข้อความชิดขวา
text-justify	กระจายข้อความให้พอดี
text-wrap	wrap ข้อความ
text-nowrap	ไม่ wrap ข้อความ
text-uppercase	ปรับเป็นตัวพิมพ์ใหญ่
text-lowercase	ปรับเป็นตัวพิมพ์เล็ก

2. Element Padding **ใช้สำหรับเติมช่องว่าง หรือเว้นช่องให้กับ** element

padding	เพิ่มช่องว่างให้กับ element ทั้งหมด
padding-top	เพิ่มช่องว่างด้านบน
padding-left	เพิ่มช่องว่างด้านซ้าย
padding-right	เพิ่มช่องว่างด้านขวา
padding-bottom	เพิ่มช่องว่างด้านล่าง
no-padding	ลบช่องว่างทั้งหมดออกไป

3. **ion-buttons property** จัดและเรียงปุ่ม (Button)

start	iOS จะชิดซ้าย ถ้า Android ตัวแรกจะชิดขวา
end	iOS จะชิดซ้าย ถ้า Android ตัวสุดท้ายจะชิดขวา
left	จัดปุ่มชิดด้านซ้าย
right	จัดปุ่มชิดด้านขวา

การตกแต่ง App โดยการเขียนทับตัวแปร Sass ของ Ionic

เราสามารถเขียนทับตัวแปร Sass ของ Ionic ที่มีอยู่แล้วได้ การใช้งานคือให้เปิดไฟล์ app/theme/app.variables.scss แล้วเขียนตัวแปรพร้อมค่าที่ต้องการ ตัวอย่างเช่น

```
// http://ionicframework.com/docs/v2/theming/

// Ionic Shared Functions
// --------------------------------------------------
// Makes Ionic Sass functions available to your App

@import "globals.core";

// App Shared Variables
// --------------------------------------------------
// To customize the look and feel of this app, you can override
// the Sass variables found in Ionic's source scss files. Setting
// variables before Ionic's Sass will use these variables rather than
// Ionic's default Sass variable values. App Shared Sass imports belong
// in the app.core.scss file and not this file. Sass variables specific
// to the mode belong in either the app.ios.scss or app.md.scss files.


// App Shared Color Variables
// --------------------------------------------------
// It's highly recommended to change the default colors
// to match your app's branding. Ionic uses a Sass map of
// colors so you can add, rename and remove colors as needed.
// The "primary" color is the only required color in the map.
// Both iOS and MD colors can be further customized if colors
// are different per mode.
```

```
$colors: (
  primary:    #387ef5,
  secondary:  #32db64,
  danger:     #f53d3d,
  light:      #f4f4f4,
  dark:       #222,
  favorite:   #69BB7B
);

$list-background-color: #fff;

$list-ios-activated-background-color: #3aff74;
$list-md-activated-background-color: #3aff74;
$checkbox-ios-background-color-on: #32db64;
$checkbox-ios-icon-border-color-on: #fff;

$checkbox-md-icon-background-color-on: #32db64;
$checkbox-md-icon-background-color-off: #fff;
$checkbox-md-icon-border-color-off: #cecece;
$checkbox-md-icon-border-color-on: #32db64;
```

ตัวแปรที่เราสามารถเขียนทับได้ดูรายละเอียดทั้งหมดได้ที่นี่ http://ionicframework.com/docs/v2/theming/overriding-ionic-variables/

> **Note:** หากลองแก้สีต่างๆแล้วอย่าลืมบันทึกไฟล์แล้วลองรันดูอีกครั้ง

จัดรูปแบบการแสดงข้อมูลด้วย Pipes

Pipes ใน Angular 2 มีไว้สำหรับ transform, filter หรือไว้จัดรูปแบบข้อมูลต่างๆ เราสามารถแทรก | เข้าไปใน html ได้เลยครับ การใช้งาน Pipes ใน Ionic นั้นมี 2 วิธีคือ

1. เรียกใช้ได้เลย คือ ใน Angular 2 จะมี Built-in pipes เราสามารถเรียกใช้ได้เลยด้วยเครื่องหมาย |
2. สร้าง custom pipe เอง แล้วเวลาใช้งานก็อ้างชื่อ pipe นั้นๆ

พื้นฐานการใช้งาน Built-in pipes ใน Angular 2

1. uppercase มีไว้แปลงตัวอักษรให้เป็นตัวพิมพ์ใหญ่ทั้งหมด

```
<p>{{ 'Coding Thailand' | uppercase }}</p>

<!– ผลลัพธ์ คือ CODING THAILAND –>
```

2. lowercase มีไว้แปลงตัวอักษรให้เป็นตัวพิมพ์เล็กทั้งหมด

```
<p>{{ 'Coding Thailand' | lowercase }}</p>

<!– ผลลัพธ์ คือ coding thailand –>
```

3. number ใช้สำหรับจัดรูปแบบของตัวเลขต่างๆ เช่น

```
<p>{{ 12345 }}</p>

<!– ผลลัพธ์ คือ '12345' –>

 <p>{{ 12345 | number }}</p>

<!– ผลลัพธ์ คือ '12,345' –>

 <p>{{ 12345 | number:'6.' }}</p>

<!– ผลลัพธ์ คือ '012,345' –>

 <p>{{ 12345 | number:'.2' }}</p>

<!– ผลลัพธ์ คือ '12,345.00' –>
```

4. percent จัดรูปแบบข้อมูลในรูปแบบ %

```
<p>{{ 0.8 | percent }}</p>

<!– ผลลัพธ์ คือ ‘80%’ –>

<p>{{ 0.8 | percent:’.3’ }}</p>

<!– ผลลัพธ์ คือ ‘80.000%’ –>
```

5. currency ใช้สำหรับจัดรูปแบบในรูปสกุลเงิน

```
<p>{{ 10.6 | currency:’EUR’ }}</p>

<!– ผลลัพธ์ คือ ‘EUR10.6’ –>

<p>{{ 10.6 | currency:’USD’:true }}</p>

<!– ผลลัพธ์ คือ ‘$10.6’ –>

<p>{{ 10.6 | currency:’USD’:true:’.2’ }}</p>

<!– ผลลัพธ์ คือ ‘$10.60’ –>
```

6. date ใช้สำหรับจัดรูปแบบวันที่และเวลา

```
<p>{{ birthday | date:’ddMMyyyy’ }}</p>

<!– ผลลัพธ์ คือ ’07/16/1986’ –>

<p>{{ birthday | date:’longDate’ }}</p>
```

<!– ผลลัพธ์ คือ ‘July 16, 1986’ –>

{{ birthday | date:’HHmmss’ }}

<!– ผลลัพธ์ คือ ’15:30:00’ –>

{{ birthday | date:’shortTime’ }}

<!– ผลลัพธ์ คือ ‘3:30 PM’ –>

Note: การใช้งาน Pipes ดูเพิ่มเติมได้ที่ https://angular.io/docs/ts/latest/guide/pipes.html

Note: ดูรายละเอียด Built-in pipes เพิ่มเติมได้ที่ https://angular.io/docs/ts/latest/api/#!?apiFilter=pipe

การสร้าง Custom Pipes ใน Ionic 2

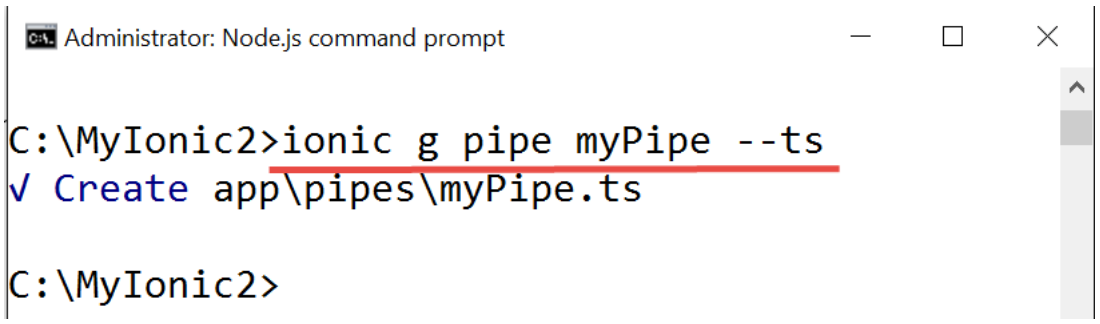
สำหรับสร้างและใช้งาน Pipes ใน Ionic 2 เราสามารถ gen code ได้เลยโดยใช้ Ionic CLI รูปแบบการใช้งานเหมือนของ Angular 2 ปกติ

{{someData | somePipe}}

รูปแบบคำสั่งในการสร้าง Pipes คือ **ionic g pipe <ชื่อ> --ts**

ขั้นตอนการสร้างและใช้งาน Pipes ใน Ionic 2 มีดังนี้

1. การสร้าง Pipe ใช้คำสั่ง **ionic g pipe myPipe --ts** แล้วกดปุ่ม enter



2. เปิดไฟล์ app\pipes\myPipe.ts

```
import { Injectable, Pipe } from '@angular/core';

@Pipe({
//ชื่อ (name) ของ pipe ห้ามมีขีดกลาง
  name: 'mypipe' //ชื่อ pipe นี้คือชื่อที่จะนำไปใช้ เช่น {{ somestring | mypipe}}
})
@Injectable()
export class MyPipe {
  /*
    Takes a value and makes it lowercase.
    การเขียน pipe จะเขียนใน method transform แล้วให้คืนค่ากลับ
    ในตัวอย่างนี้คือการแปลง string ไปเป็นตัวพิมพ์ใหญ่ทั้งหมด
     ถ้าหากอยากแปลงแบบไหนก็เขียนได้ตรงนี้
   */
  transform(value: string, args: any[]) {
    value = value + ''; // make sure it's a string
    return value.toUpperCase();
  }
}
```

3. การเรียกนำ Pipe เข้ามาในเพจ จะยกตัวอย่างเพจ movies ให้เปิดไฟล์ app\pages\movies\movies.ts เพิ่มโค้ดดังนี้

```
import { Component } from '@angular/core';
import { NavController, Loading } from 'ionic-angular';
import { Movies } from '../../providers/movies/movies';
//import คลาส Pipe ชื่อ เข้ามาใช้งาน
import { MyPipe } from '../../pipes/myPipe';

@Component({
  pipes: [MyPipe],
  providers: [Movies],
```

```
  templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
  items: any = null;

  constructor(private nav: NavController, private movie: Movies) {
    this.movie = movie;
    this.showData(); //เรียกใช้ method showData()
  }

  showData() {
    //สร้าง Loading พร้อมกำหนดข้อความ
    let loading = Loading.create({
      content: "รอกสักครู่...",
      spinner: 'dots',
      duration: 3000
    });

    //แสดง Loading หากโหลดข้อมูลเรียบร้อยแล้ว ก็ให้หายไป (dismiss)
    this.nav.present(loading).then(() => {
        this.items = this.movie.load();
        loading.dismiss();
    });
  }

}
```

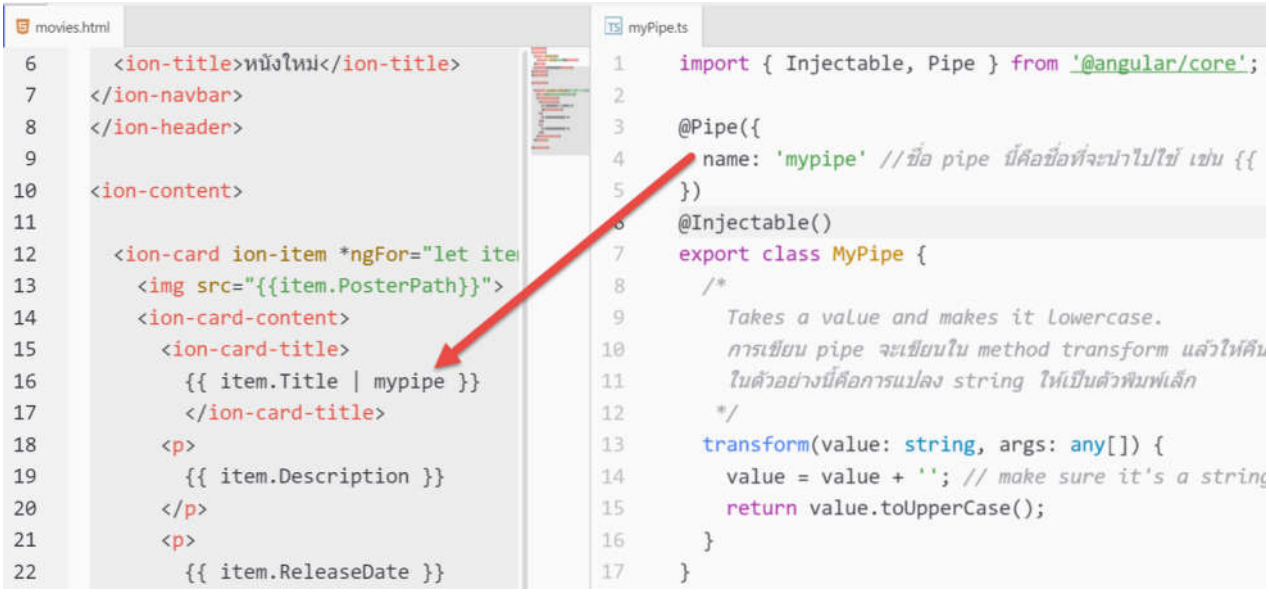
4. การเรียกใช้งาน Pipe ที่ HTML เปิดไฟล์ app\pages\movies\movies.html เขียนโค้ด ดังนี้

```
<ion-header>
<ion-navbar>
  <button menuToggle>
```

```
    <ion-icon name="menu"></ion-icon>
  </button>
  <ion-title>หนังใหม่</ion-title>
</ion-navbar>
</ion-header>
<ion-content>
  <ion-card ion-item *ngFor="let item of items | async">
    <img src="{{item.PosterPath}}">
    <ion-card-content>
      <ion-card-title>
        {{ item.Title | mypipe }}
        </ion-card-title>
      <p>
        {{ item.Description }}
      </p>
      <p>
        {{ item.ReleaseDate }}
      </p>
    </ion-card-content>
  </ion-card>


</ion-content>
```

5. บันทึกไฟล์ แล้วทดสอบ App อีกครั้ง ชื่อภาพยนตร์จะกลายเป็นตัวพิมพ์ใหญ่ทั้งหมด

ข้อควรระวัง ในการใช้ Pipe คือ ตอนตั้งชื่อ กับตอนเรียกใช้ ต้องเขียนให้ตรงกันด้วยครับ



movies.html

```
6     <ion-title>หนังใหม่</ion-title>
7   </ion-navbar>
8   </ion-header>
9
10  <ion-content>
11
12    <ion-card ion-item *ngFor="let ite
13      <img src="{{item.PosterPath}}">
14      <ion-card-content>
15        <ion-card-title>
16          {{ item.Title | mypipe }}
17          </ion-card-title>
18        <p>
19          {{ item.Description }}
20        </p>
21        <p>
22          {{ item.ReleaseDate }}
```

myPipe.ts

```
1   import { Injectable, Pipe } from '@angular/core';
2
3   @Pipe({
4     name: 'mypipe' //ชื่อ pipe นี้คือชื่อที่จะนำไปใช้ เช่น {{
5   })
6   @Injectable()
7   export class MyPipe {
8     /*
9       Takes a value and makes it lowercase.
10      การเขียน pipe จะเขียนใน method transform แล้วให้คืน
11      ในตัวอย่างนี้คือการแปลง string ให้เป็นตัวพิมพ์เล็ก
12     */
13    transform(value: string, args: any[]) {
14      value = value + ''; // make sure it's a string
15      return value.toUpperCase();
16    }
17  }
```

บทที่ 11 การเตรียม Resources และ Publish App

หลังจากที่เราได้สร้าง App เรียบร้อย ขั้นตอนต่อไปคือการเตรียม App ของเราเพื่ออัพโหลดขึ้น Store ครับ มาดูกันว่าเรามีอะไรที่ต้องทำกันบ้าง

การเปิด Production Mode

ปกติแล้วขณะที่เราพัฒนา App อยู่โหมดของการพัฒนาจะเป็น development หากเราพัฒนาเสร็จแล้วต้องการนำ App ขึ้น Store ก็อย่าลืมกำหนดโหมดให้เป็น production ครับ การกำหนดโหมดการพัฒนานี้เป็นของ Angular 2 เราสามารถกำหนดโหมดได้ที่ไฟล์ **app\app.ts** โดยกำหนดที่ method ionicBootstrap() (ด้านล่างสุด) ดังนี้

```
ionicBootstrap(MyApp, [], {
 prodMode: true //true คือ production mode
});
```

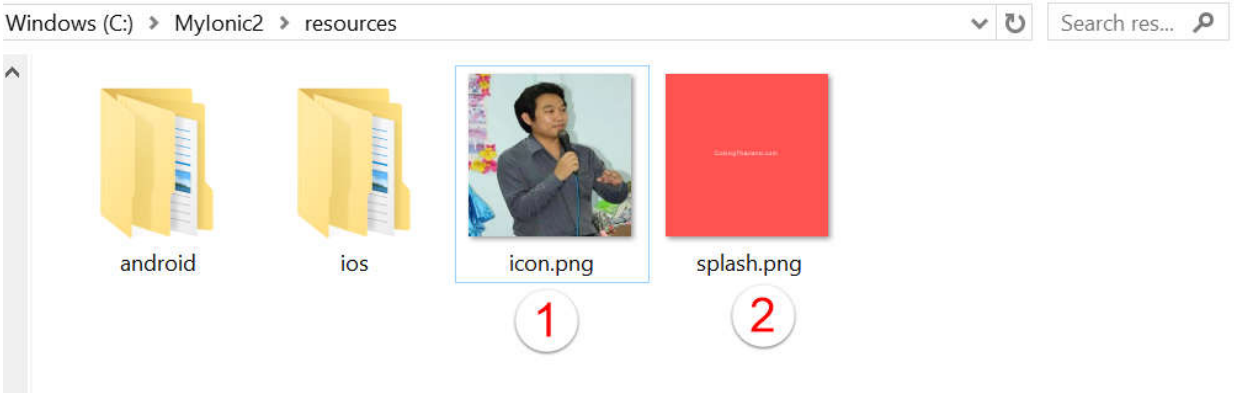
แน่นอนหากต้องการปรับกลับมาเป็น development mode ก็ให้ **prodMode: false**

> **Note:** ดูการ Config อื่นๆ นอกจาก prodMode ได้ที่ http://ionicframework.com/docs/v2/api/config/Config/

การออกแบบไอคอน และ splash screen ให้กับ App

การออกแบบไอคอนให้กับ App แนะนำขนาดภาพอย่างน้อยขนาด 1024 x 1024 มีชนิดภาพเป็น PNG ครับ วิธีการสร้างไอคอนก็ง่ายมากเมื่อออกแบบเสร็จแล้ว ให้ copy ไอคอนไปวางทับไฟล์ icon.png ในโฟลเดอร์ resources ได้เลย

สำหรับการออกแบบ splash screen แนะนำขนาดภาพ 2208 x 2208 ชนิดภาพ PNG เมื่อออกแบบเสร็จแล้วก็ให้ไปวางทับไฟล์ splash.png ในโฟลเดอร์ resources เช่นเดียวกัน

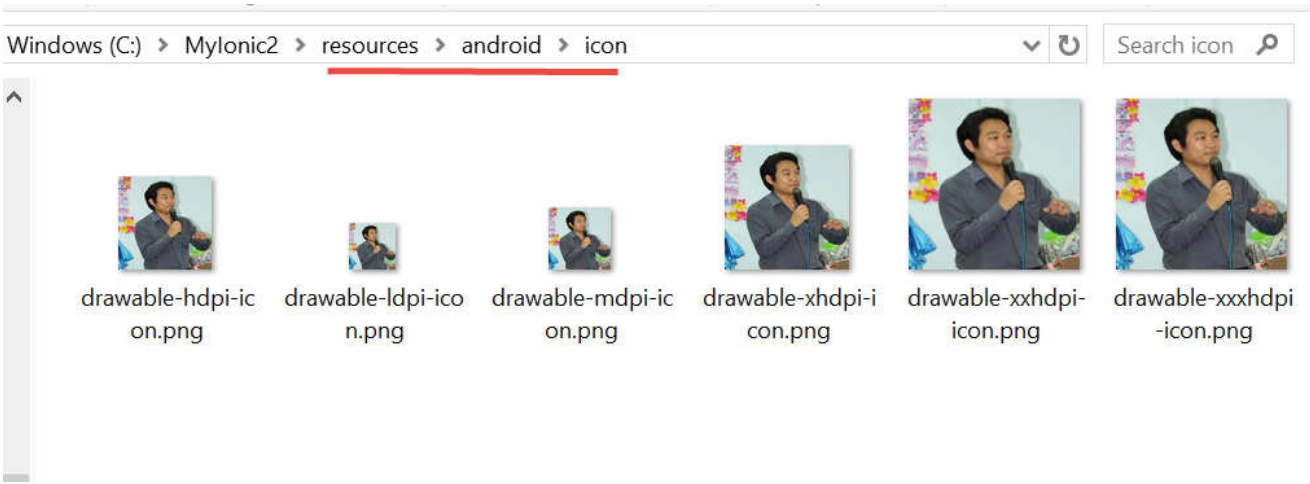


หลังจากที่เราได้ไอคอน และ splash แล้ว ต่อมาให้ใช้คำสั่ง **ionic resources android ios** เพื่อให้ Ionic จัดการสร้างไอคอน และ splash screen ให้เราทั้งในส่วนของ Android และ iOS

เปิด command prompt ขึ้นมาแล้วสั่งรัน **ionic resources android ios** แล้วกด Enter ครับ

```
Administrator: Node.js command prompt

c:\MyIonic2>ionic resources android ios
```

เราจะเห็นว่า Ionic ได้สร้างไฟล์ไอคอน และ splash screen ให้เราเรียบร้อย



ตั้งค่า Bundle ID และ App Name สำหรับ App ที่ Build ด้วย Ionic

ต่อมาขั้นตอนนี้เป็นตั้ง ID ซึ่งจะไม่ซ้ำกับ App อื่นๆในโลกนี้ หลักการตั้งเราอาจตั้งในรูปแบบเช่น com.yourname.yourproject ก็ได้ นอกจากนั้นก็ตั้งชื่อ App พร้อมเลข version ด้วยก็ได้

เปิด **ไฟล์** config.xml ขึ้นมาแล้วแก้ไขรายละเอียดให้เป็นของตัวเอง ตัวอย่างในภาพ

การลดขนาดภาพ

เพื่อให้ App เราโหลดเร็วที่สุด สิ่งสำคัญต่อมาคือ การลดขนาดของรูปภาพทั้งหมดใน App ของเราครับ แนะนำให้ทำด้วย อาจใช้เครื่องมือนี้ได้เลย http://tinypng.com/

หลังจากเสร็จขั้นตอนด้านบนนี้ แนะนำให้รัน App ดูใน Emulator หรือ USB อีกครั้งครับว่าไอคอน และ splash screen ของเรามาหรือยัง

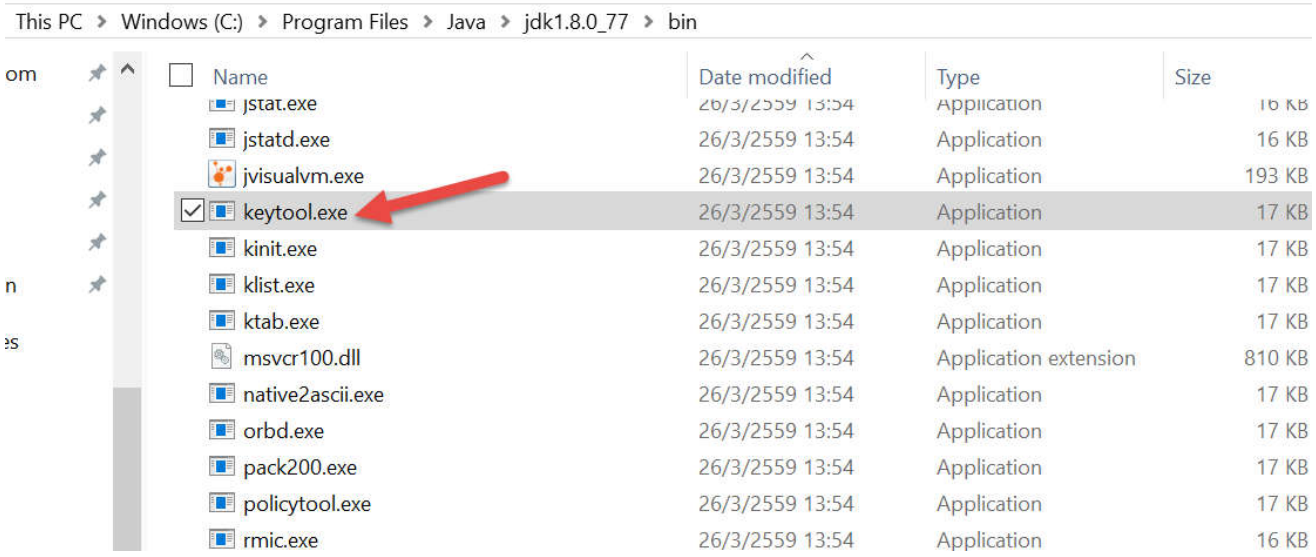
การ Sign applications

หลังจากเสร็จขั้นตอนต่างๆแล้ว Mobile App ที่เราพัฒนาขึ้น ไม่ว่าจะเป็น iOS หรือ Android จะต้องมีการ sign app เสียก่อน ถ้าเป็น iOS เราจะต้องสร้างไฟล์นามสกุล .p12 ถ้าใช้ MAC ก็ใช้ XCode ทำได้เลย แต่ถ้าเราไม่มี MAC เราสามารถใช้ OpenSSL แทนได้ครับ ส่วน Android ก็คล้ายกันจะต้องสร้างไฟล์ .keystore ก่อน แน่นอนต้องติดตั้ง Android SDK ให้เรียบร้อย

Note: หนังสือเล่มนี้จะเน้นไปที่ Android เป็นหลัก ถ้าสงสัยเรื่องการ sign app สำหรับ iOS สามารถปรึกษาได้ทีหลังครับ

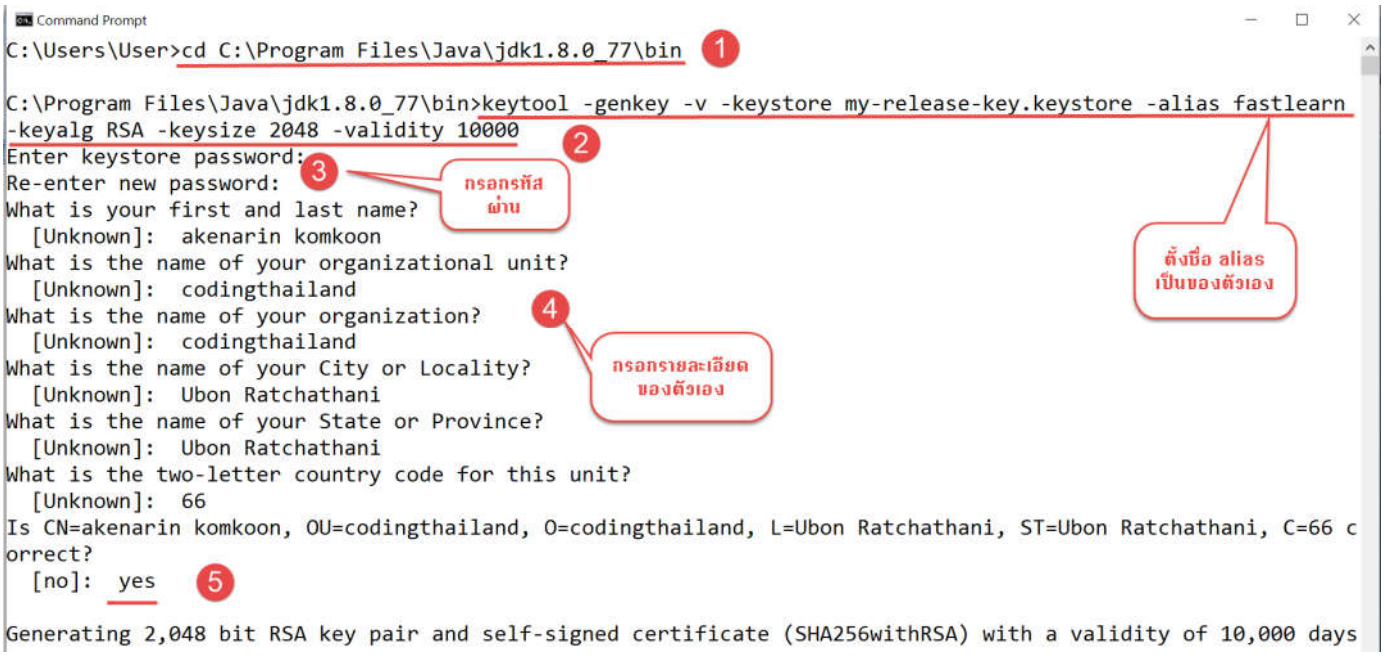
การ Sign Android Applications

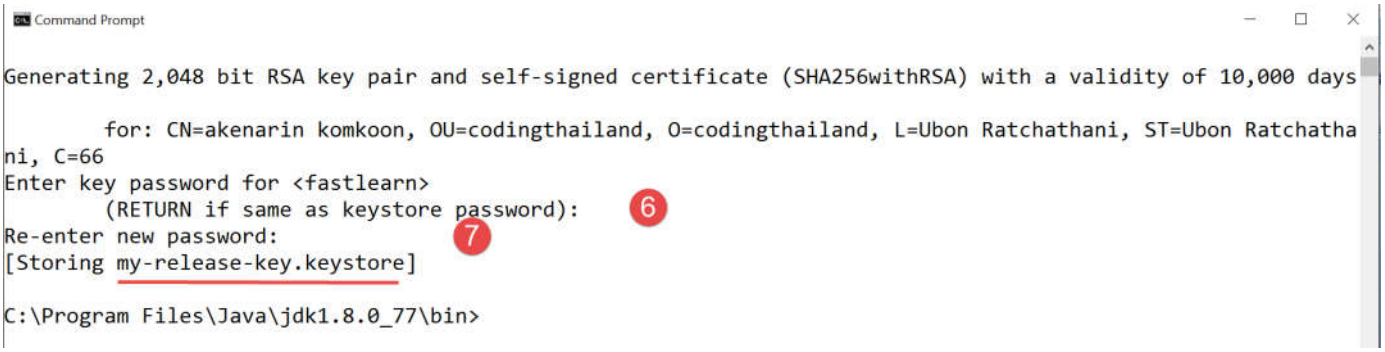
ก่อนที่จะอัพโหลดเพื่อไป Build App เราจะต้อง sign app เราก่อน ใน Android คือ เราต้องสร้าง**ไฟล์ .keystore** ครับ ในการ sign android app นั้น เราจะใช้เครื่องมือที่ชื่อว่า **keytool** ซึ่งจะมีติดมาอยู่แล้วกับ JAVA JDK ในเครื่องของเรา



Warning: โฟลเดอร์ bin ของ jdk ควรสามารถเขียนได้ (ไม่ read-only)

การ Sign app เราจะใช้ command line ของ Windows ตามขั้นตอนดังนี้





คำสั่งสร้าง key

keytool -genkey -v -keystore my-release-key.keystore -alias ชื่อตั้งเอง -keyalg RSA -keysize 2048 -validity 10000

สำคัญมาก! -alias ให้ตั้งชื่อเป็นของเราเองครับ ห้ามตั้งซ้ำกับผมนะ! แล้วจำชื่อนี้ไว้ด้วยครับ เราต้องใช้ตอน build

เสร็จแล้วเราจะได้ไฟล์ .keystore ชื่อว่า **my-release-key.keystore** เรียบร้อยครับ พร้อมนำไฟล์นี้ไปอัพโหลด และ Build ต่อไป

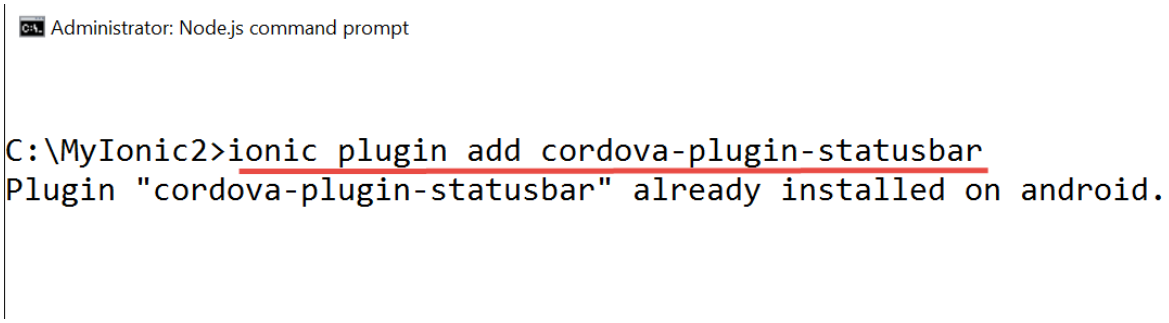
การ BUILD iOS และ Android App ด้วย PHONEGAP BUILD

หลังจากที่เราได้ Sign app เรียบร้อยก็มาถึงขั้นตอนสำคัญคือการ BUILD APP และนำไฟล์ที่ได้จากการ BUILD นี้ไปอัพโหลดขึ้น Store นั่นเอง โดยปกติแล้วหากเราจะ Build app ตัว iOS หรือ Android ก็สามารถ Build ได้เช่นกัน แต่ที่แนะนำตัวนี้เพราะเราสามารถ Build App ได้ทั้ง iOS, Android, Windows ได้ภายในครั้งเดียว และคนที่ไม่มี MAC ก็สามารถใช้ได้ครับ

ขั้นตอนการ Build App ด้วย Phonegap Build

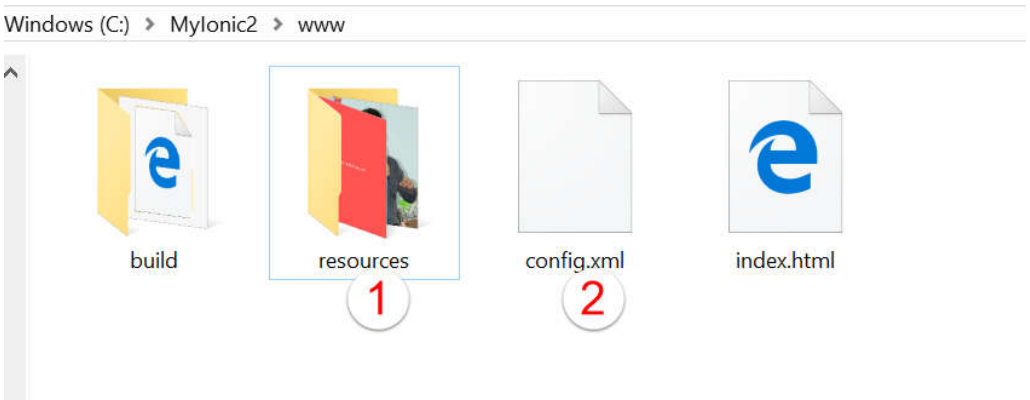
1. **ติดตั้ง plugin ที่ทุกโปรเจคควรต้องมี** ได้แก่

   1.1 plugin statusbar ใช้คำสั่ง **ionic plugin add cordova-plugin-statusbar** แล้วกด enter

   1.2 plugin crosswalk webview ใช้คำสั่ง **ionic plugin add cordova-plugin-crosswalk-webview** แล้วกด enter ประโยชน์ของ plugin ตัวนี้คือช่วยให้ App ที่เราสร้างจาก Ionic สามารถเข้ากันได้กับ Android เวอร์ชันเก่า พร้อมทั้งช่วยเราปรับปรุงประสิทธิภาพของ App ด้วย

2. Copy โฟลเดอร์ **resources** และไฟล์ **config.xml** มาไว้ในโฟลเดอร์ **www**



3. เปิดไฟล์ **www\config.xml** ขึ้นมาเพื่อแก้ไขไฟล์ เนื่องจากในหัวข้อนี้เราจะ build ด้วย Phonegap Build จึงจำเป็นต้องแก้ไขไฟล์ config.xml ใหม่ทั้งหมด โดยให้**ลบของเดิมทิ้ง** แล้ว copy **โค้ดนี้วางแทน อย่าลืมแก้รายละเอียดเป็นของตัวเองด้วยครับ)**

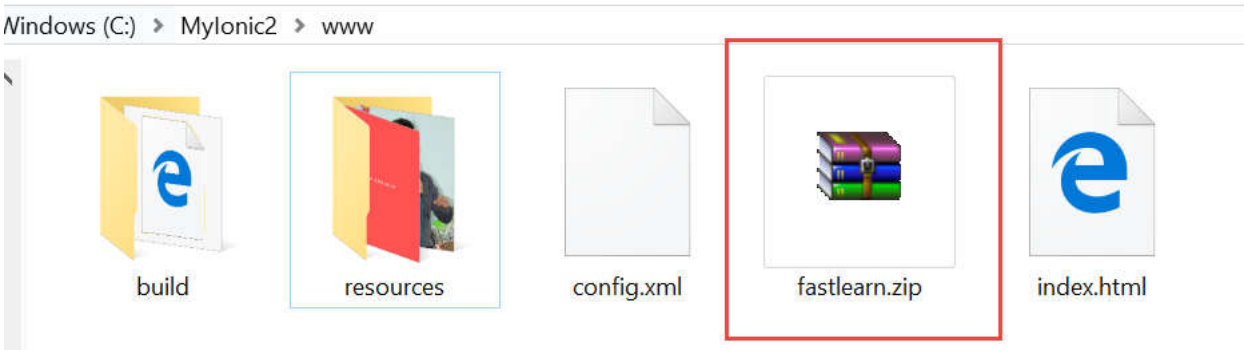
```
<?xml version="1.0" encoding="UTF-8" ?>
<widget xmlns   = "http://www.w3.org/ns/widgets"
   xmlns:gap   = "http://phonegap.com/ns/1.0"
    id            = "com.codingthailand.fastlearn"
   versionCode = "10"
   version     = "1.0.0" >

   <!-- versionCode is optional and Android only -->
    <name>Fast Learn</name>
   <description>
      Fast Learn by Coidngthailand
   </description>
   <author href="http://www.codingthailand.com" email="codingthailand@gmail.com.com">akenarin
komkoon</author>

   <plugin name="cordova-sqlite-storage" source="npm" spec="~0.8.5"/>
         <plugin name="com.phonegap.plugin.statusbar" spec="1.1.0" source="pgb" />
   <plugin name="org.apache.cordova.splashscreen" source="pgb" />
   <plugin name="com.indigoway.cordova.whitelist.whitelistplugin" source="pgb"/>

   <icon src="resources/android/icon/drawable-ldpi-icon.png" gap:platform="android" gap:qualifier="ldpi"/>
   <icon src="resources/android/icon/drawable-mdpi-icon.png" gap:platform="android" gap:qualifier="mdpi"/>
   <icon src="resources/android/icon/drawable-hdpi-icon.png" gap:platform="android" gap:qualifier="hdpi"/>
   <icon src="resources/android/icon/drawable-xhdpi-icon.png" gap:platform="android" gap:qualifier="xhdpi"/>
   <icon src="resources/android/icon/drawable-xxhdpi-icon.png" gap:platform="android" gap:qualifier="xxhdpi"/>
   <icon src="resources/android/icon/drawable-xxxhdpi-icon.png" gap:platform="android" gap:qualifier="xxxhdpi"/>
   <icon src="resources/ios/icon/icon.png" gap:platform="ios" width="57" height="57"/>
   <icon src="resources/ios/icon/icon@2x.png" gap:platform="ios" width="114" height="114"/>
```

```
    <icon src="resources/ios/icon/icon-40.png" gap:platform="ios" width="40" height="40"/>
    <icon src="resources/ios/icon/icon-40@2x.png" gap:platform="ios" width="80" height="80"/>
    <icon src="resources/ios/icon/icon-50.png" gap:platform="ios" width="50" height="50"/>
    <icon src="resources/ios/icon/icon-50@2x.png" gap:platform="ios" width="100" height="100"/>
    <icon src="resources/ios/icon/icon-60.png" gap:platform="ios" width="60" height="60"/>
    <icon src="resources/ios/icon/icon-60@2x.png" gap:platform="ios" width="120" height="120"/>
    <icon src="resources/ios/icon/icon-60@3x.png" gap:platform="ios" width="180" height="180"/>

    <preference name="prerendered-icon" value="true" />
    <preference name="target-device" value="universal" />
    <preference name="android-windowSoftInputMode" value="stateAlwaysHidden" />

    <icon src="resources/ios/icon/icon-72.png" gap:platform="ios" width="72" height="72"/>
    <icon src="resources/ios/icon/icon-72@2x.png" gap:platform="ios" width="144" height="144"/>
    <icon src="resources/ios/icon/icon-76.png" gap:platform="ios" width="76" height="76"/>
    <icon src="resources/ios/icon/icon-76@2x.png" gap:platform="ios" width="152" height="152"/>
    <icon src="resources/ios/icon/icon-small.png" gap:platform="ios" width="29" height="29"/>
    <icon src="resources/ios/icon/icon-small@2x.png" gap:platform="ios" width="58" height="58"/>
    <gap:splash src="resources/android/splash/drawable-land-ldpi-screen.png" gap:platform="android"
gap:qualifier="land-ldpi"/>
    <gap:splash src="resources/android/splash/drawable-land-mdpi-screen.png" gap:platform="android"
gap:qualifier="land-mdpi"/>
    <gap:splash src="resources/android/splash/drawable-land-hdpi-screen.png" gap:platform="android"
gap:qualifier="land-hdpi"/>
    <gap:splash src="resources/android/splash/drawable-land-xhdpi-screen.png" gap:platform="android"
gap:qualifier="land-xhdpi"/>
    <gap:splash src="resources/android/splash/drawable-land-xxhdpi-screen.png" gap:platform="android"
gap:qualifier="land-xxhdpi"/>
    <gap:splash src="resources/android/splash/drawable-land-xxxhdpi-screen.png" gap:platform="android"
gap:qualifier="land-xxxhdpi"/>
    <gap:splash src="resources/android/splash/drawable-port-ldpi-screen.png" gap:platform="android"
gap:qualifier="port-ldpi"/>
```

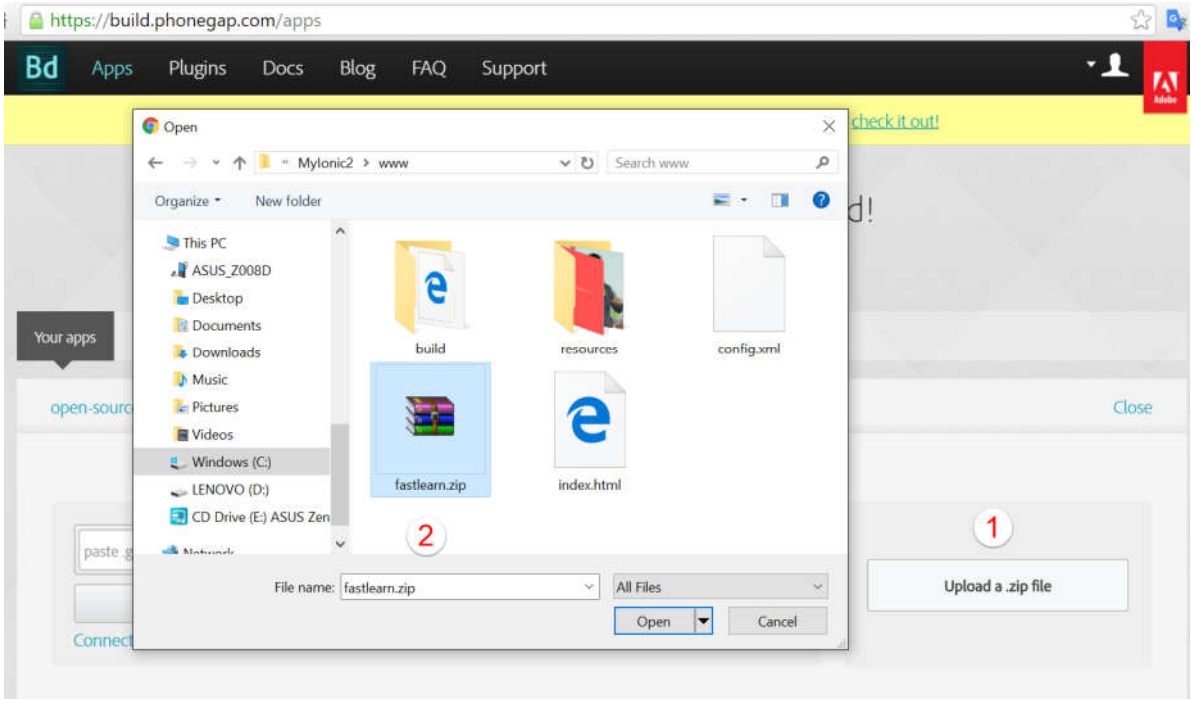
```
    <gap:splash src="resources/android/splash/drawable-port-mdpi-screen.png" gap:platform="android"
gap:qualifier="port-mdpi"/>
    <gap:splash src="resources/android/splash/drawable-port-hdpi-screen.png" gap:platform="android"
gap:qualifier="port-hdpi"/>
    <gap:splash src="resources/android/splash/drawable-port-xhdpi-screen.png" gap:platform="android"
gap:qualifier="port-xhdpi"/>
    <gap:splash src="resources/android/splash/drawable-port-xxhdpi-screen.png" gap:platform="android"
gap:qualifier="port-xxhdpi"/>
    <gap:splash src="resources/android/splash/drawable-port-xxxhdpi-screen.png" gap:platform="android"
gap:qualifier="port-xxxhdpi"/>
    <gap:splash src="resources/ios/splash/Default-568h@2x~iphone.png" gap:platform="ios" width="640"
height="1136"/>
    <gap:splash src="resources/ios/splash/Default-667h.png" width="750" gap:platform="ios" height="1334"/>
    <gap:splash src="resources/ios/splash/Default-736h.png" width="1242" gap:platform="ios" height="2208"/>

    <gap:splash src="resources/ios/splash/Default-Landscape-736h.png" gap:platform="ios" width="2208"
height="1242"/>
    <gap:splash src="resources/ios/splash/Default-Landscape@2x~ipad.png" gap:platform="ios" width="2048"
height="1536"/>
    <gap:splash src="resources/ios/splash/Default-Landscape~ipad.png" gap:platform="ios" width="1024"
height="768"/>
    <gap:splash src="resources/ios/splash/Default-Portrait@2x~ipad.png" gap:platform="ios" width="1536"
height="2048"/>
    <gap:splash src="resources/ios/splash/Default-Portrait~ipad.png" gap:platform="ios" width="768" height="1024"/>
    <gap:splash src="resources/ios/splash/Default@2x~iphone.png" gap:platform="ios" width="640" height="960"/>
    <gap:splash src="resources/ios/splash/Default~iphone.png" gap:platform="ios" width="320" height="480"/>

        <!-- These are all equivalent -->
        <preference name="xwalkCommandLine" value="--disable-pull-to-refresh-effect"/>
        <preference name="xwalkMode" value="embedded"/>
        <preference name="xwalkMultipleApk" value="true"/>
```

```
    <!-- Default Icon and Splash -->
    <icon src="resources/icon.png" />
    <gap:splash src="resources/splash.png" />
    <access origin="*"/>

</widget>
```

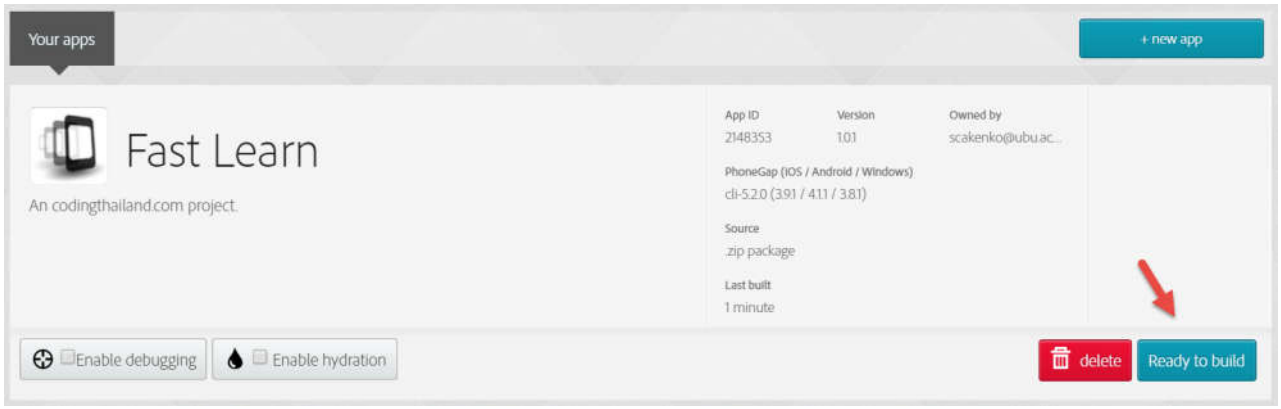
4. จากนั้นให้ zip ไฟล์ทั้งหมดใน www แล้วตั้งชื่อตามต้องการ



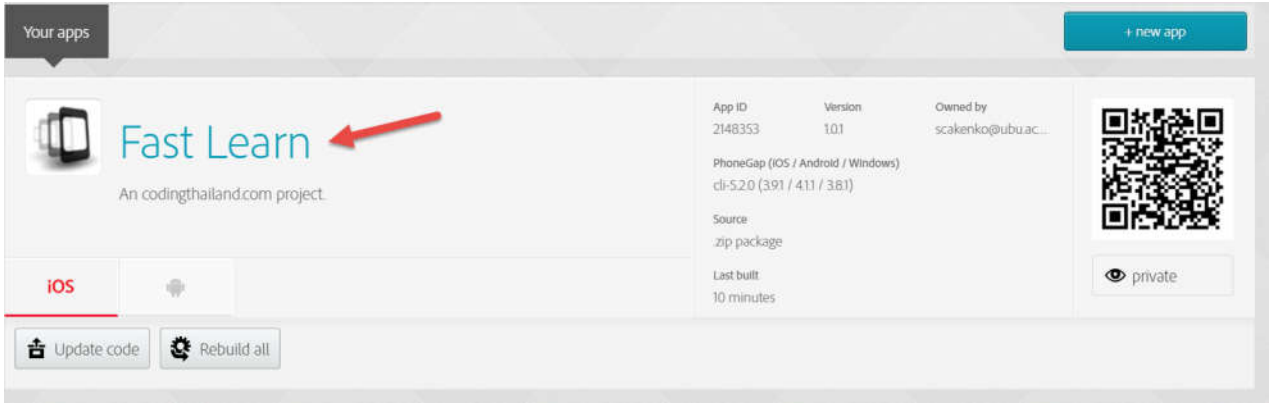
5. เข้าเว็บ https://build.phonegap.com ให้สมัครสมาชิกและล็อกอินให้เรียบร้อย
6. กลับมาที่หน้า Apps หรือกดปุ่ม new app **เลือก** Upload a .zip file แล้วเลือกไฟล์ที่เราได้ zip แล้วในข้อ 3



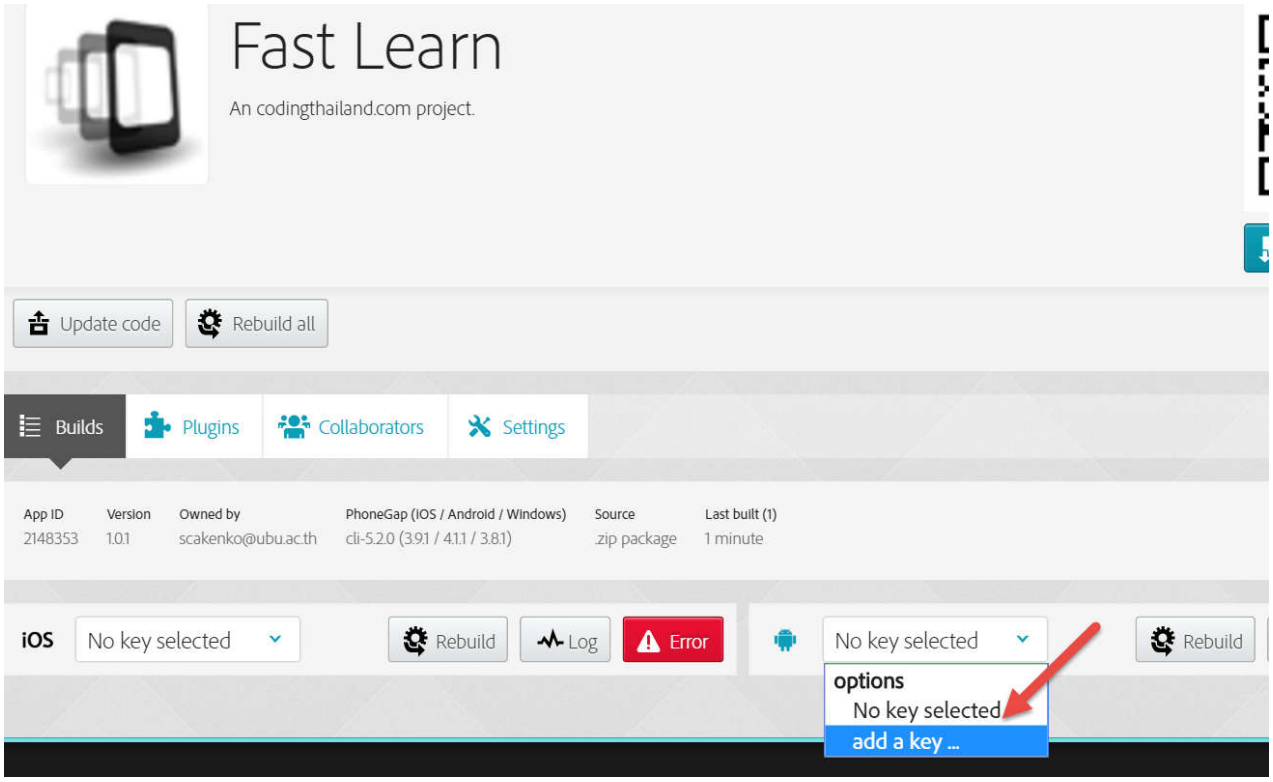
7. คลิกที่ปุ่ม Ready to build



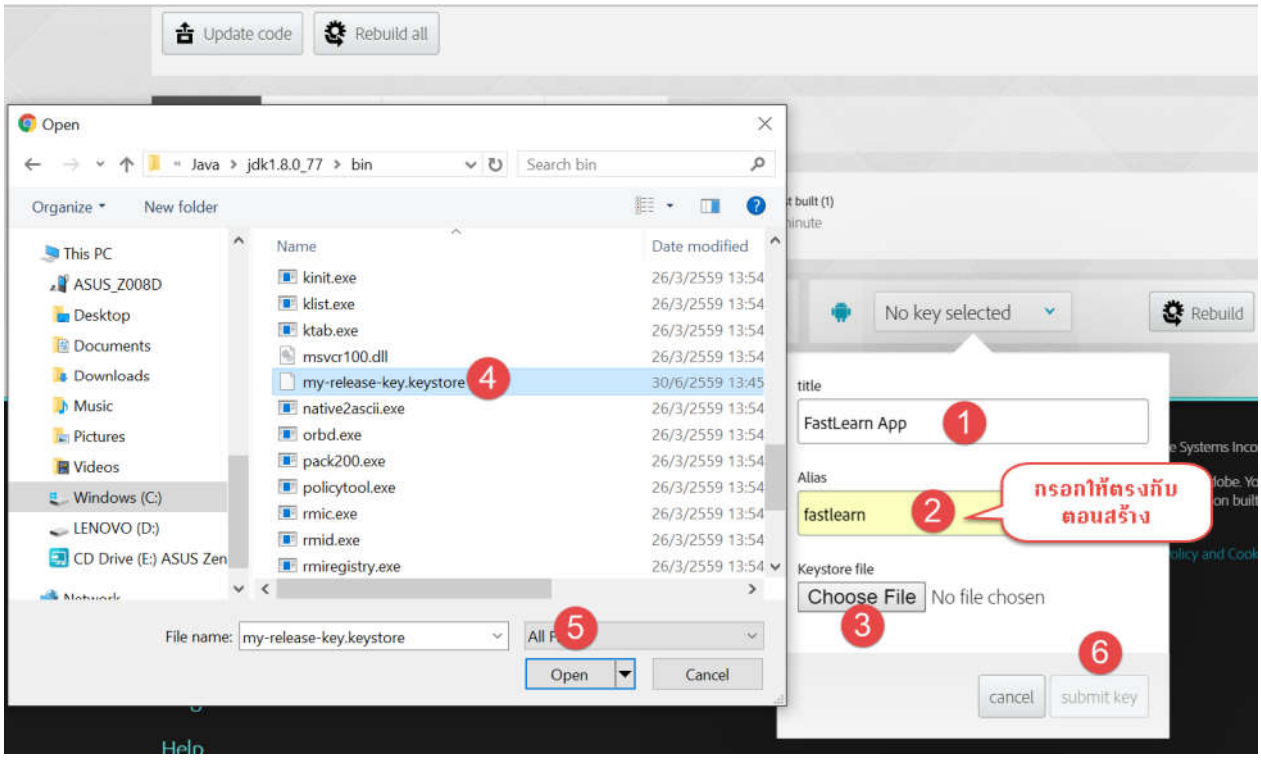
8. คลิกที่ชื่อโปรเจค



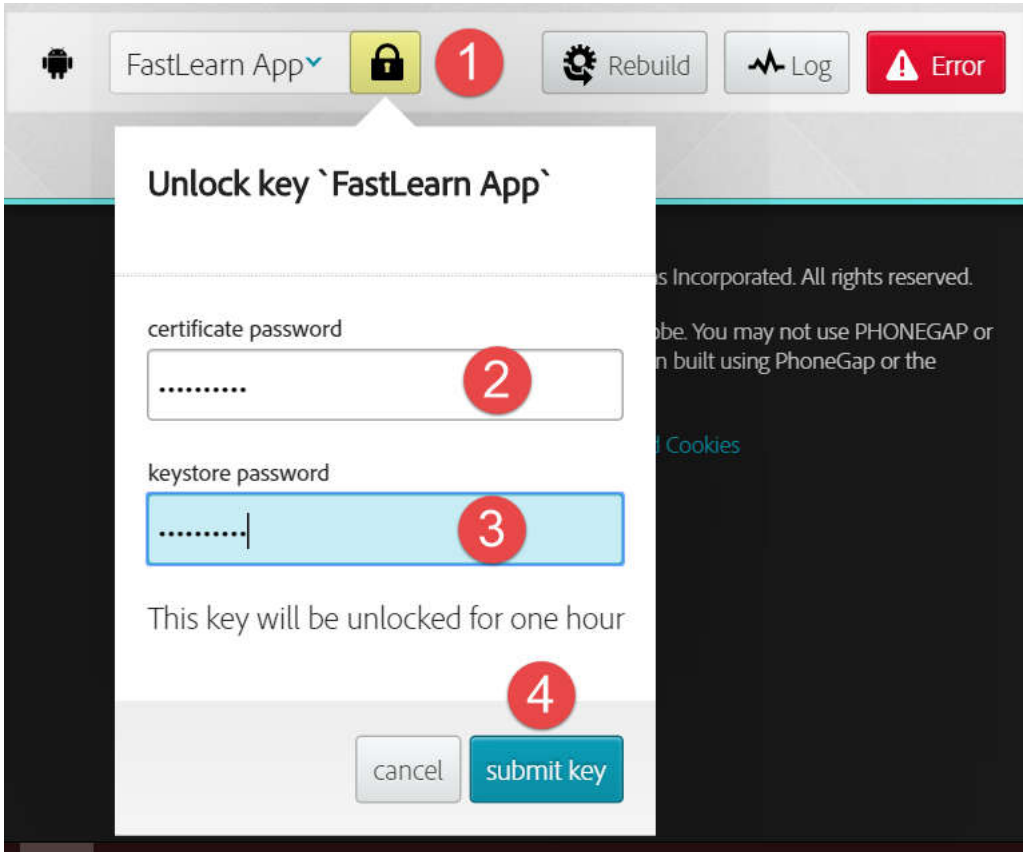
9. คลิกเลือก add a key



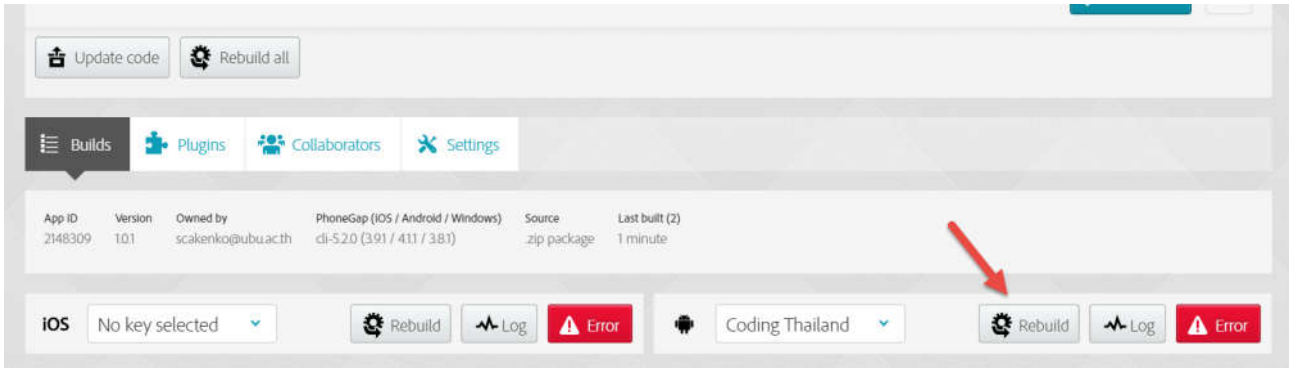
10. กรอก title (อะไรก็ได้หรือชื่อ app) กรอก alias **(จะต้องตรงกับตอนสร้าง .keystore)** แล้วเลือกไฟล์ .keystore ที่เราได้สร้างไว้แล้วกด submit key



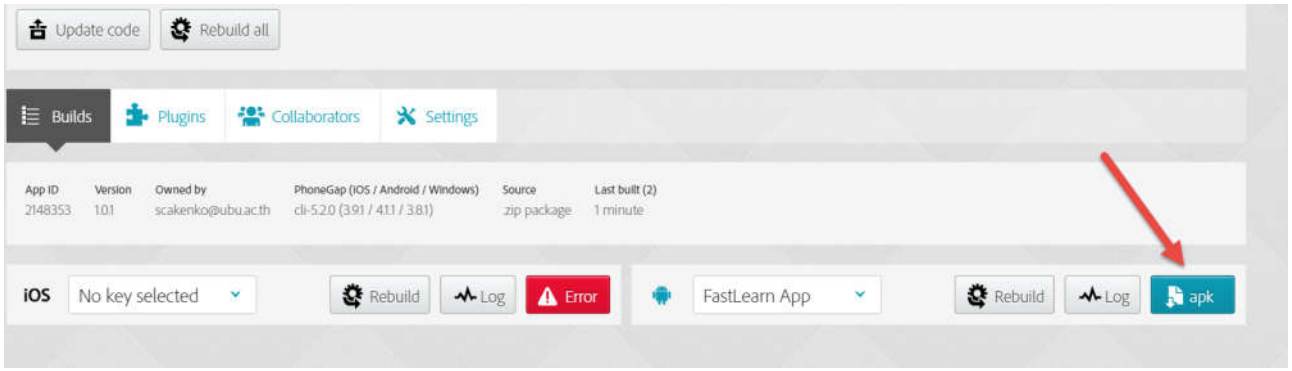
11. ใส่รหัสผ่านไฟล์ .keystore ของเรา (**รหัสนี้เป็นรหัสที่เรากรอกตอนสร้างไฟล์ .keystore** นั่นเอง) เสร็จแล้วกด submit key



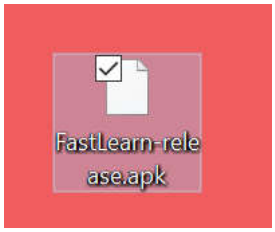
12. คลิกปุ่ม Rebuild รอสักครู่



13. จากนั้นเราจะได้ไฟล์ .apk พร้อมอัพโหลดขึ้น Google Play (Store) เรียบร้อย! กดดาวน์โหลดไฟล์ .apk มาไว้ในเครื่องเราเลยครับ



14. หน้าตาของไฟล์ .apk เป็นแบบนี้



Note: หากมีการแก้ไขโค้ดแล้วอยากอัพโหลดเพื่อไป build ใหม่ก็เลือกปุ่ม Update code ได้เลยครับ เสร็จแล้วก็กด Rebuild

การ Build Android App ด้วย Ionic CLI (iOS ใช้วิธีการเดียวกัน)

มาดูอีก 1 วิธีในการ Build App กันครับ สามารถเลือกอย่างใดอย่างหนึ่งได้ จะใช้ PHONEGAP BUILD ในหัวข้อที่แล้ว หรือจะใช้ Ionic CLI ในหัวข้อนี้ก็ได้ เลือกได้ครับ

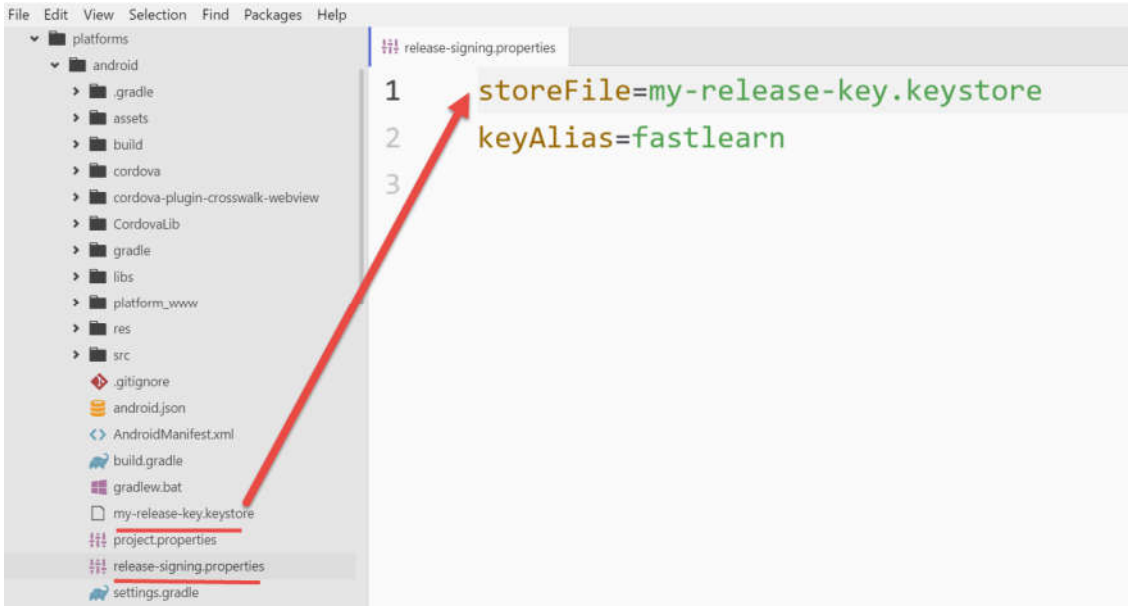
ขั้นตอนการ Build Android App ด้วยคำสั่ง Ionic CLI มีขั้นตอน ดังนี้

1. Copy ไฟล์ **.keystore (my-release-key.keystore)** ที่สร้างไว้แล้ว ไปวางไว้ที่โฟลเดอร์ **platforms\android**
2. ในโฟลเดอร์ **platforms\android** นี้ ให้สร้างไฟล์ชื่อว่า **release-signing.properties**
3. เปิดไฟล์ **platforms\android\release-signing.properties** เขียนกำหนดค่า ดังนี้

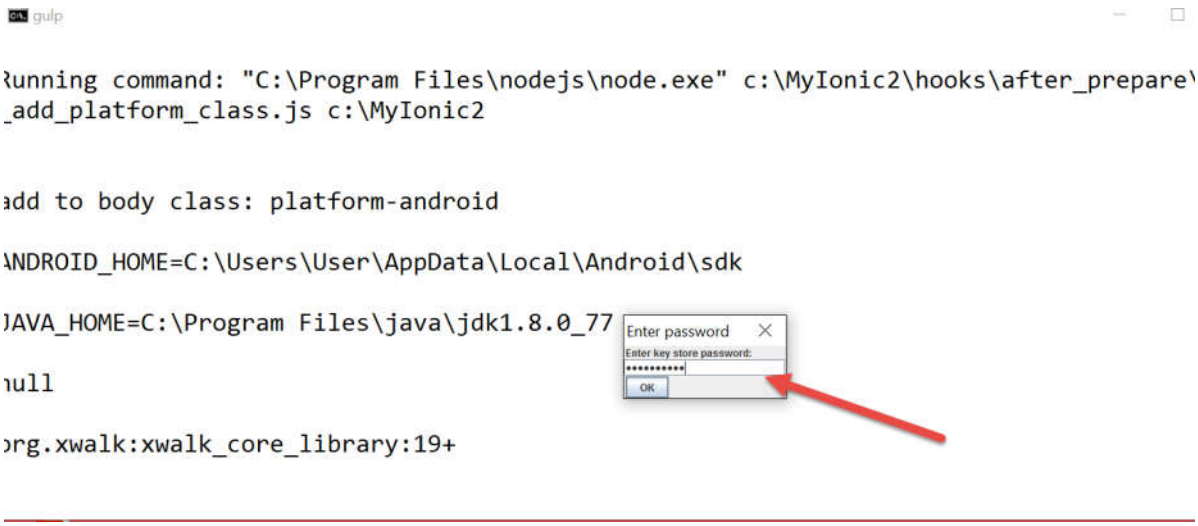
storeFile=**my-release-key.keystore**

keyAlias=**fastlearn**

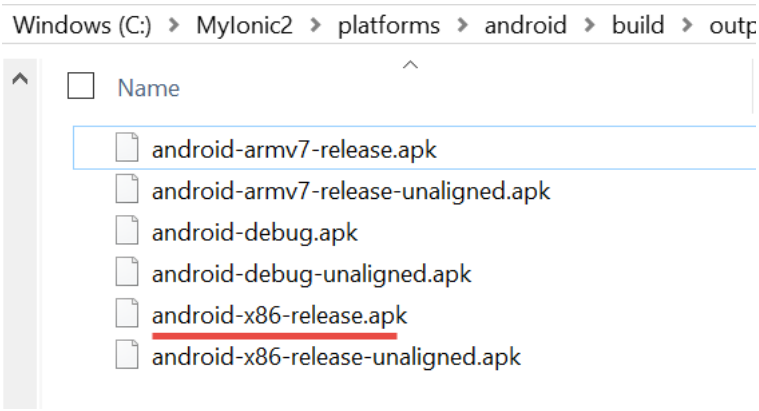
Note: ตรง fastlearn อย่าลืมแก้เป็น alias ของตัวเองนะครับ



4. บันทึกไฟล์ แล้วรันคำสั่ง **ionic build android --release** แล้วกด enter
5. กรอกรหัสผ่านของไฟล์ .keystore (2 ครั้ง)



6. รอสักครู่ เมื่อเสร็จแล้วสามารถตรวจสอบไฟล์ .apk ได้ที่โฟลเดอร์ **platforms\android\build\outputs\apk**



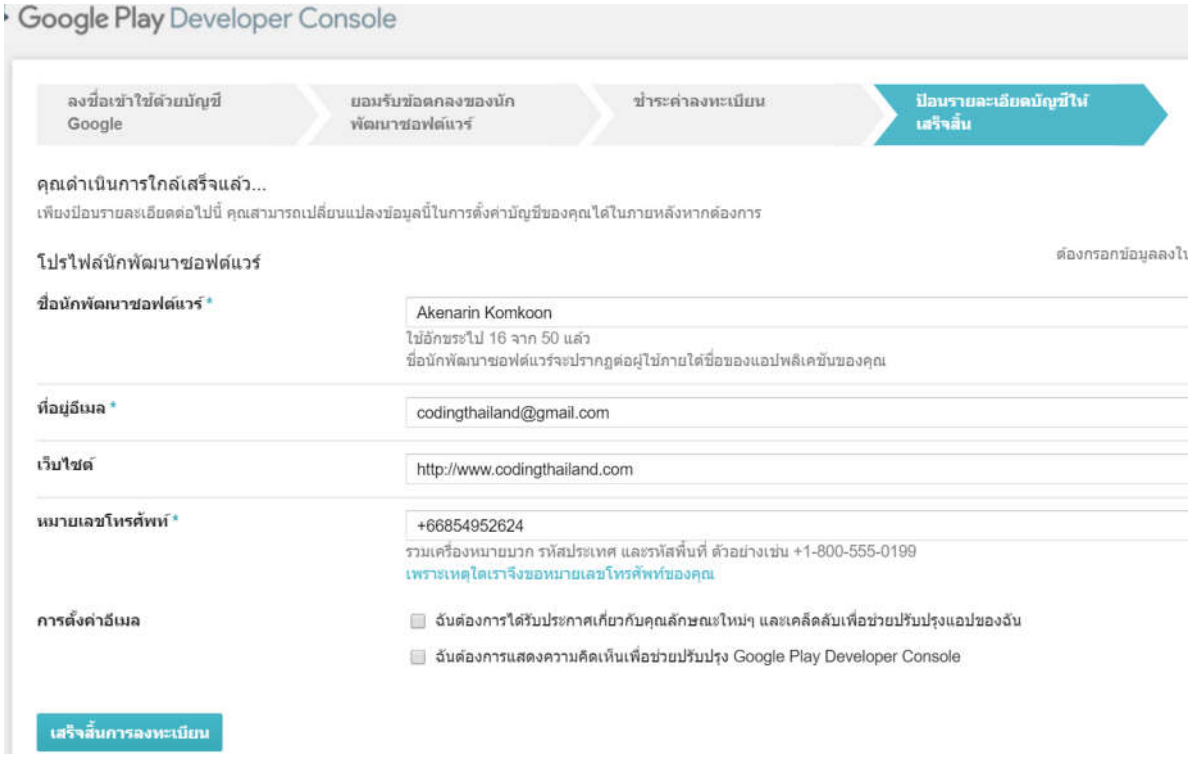
7. เสร็จเรียบร้อยแล้วครับ พร้อมอัพโหลดขึ้น Store!

Note: ถ้าไฟล์ใหญ่ เป็นเพราะเราใช้ plug-in crosswalk ก็แนะนำให้ใช้ Crosswalk Lite ได้ครับขนาดจะลดลงมา ดูได้ที่ https://crosswalk-project.org/blog/crosswalk-lite-10.html และอย่าลืม optimize รูปภาพด้วยนะครับ

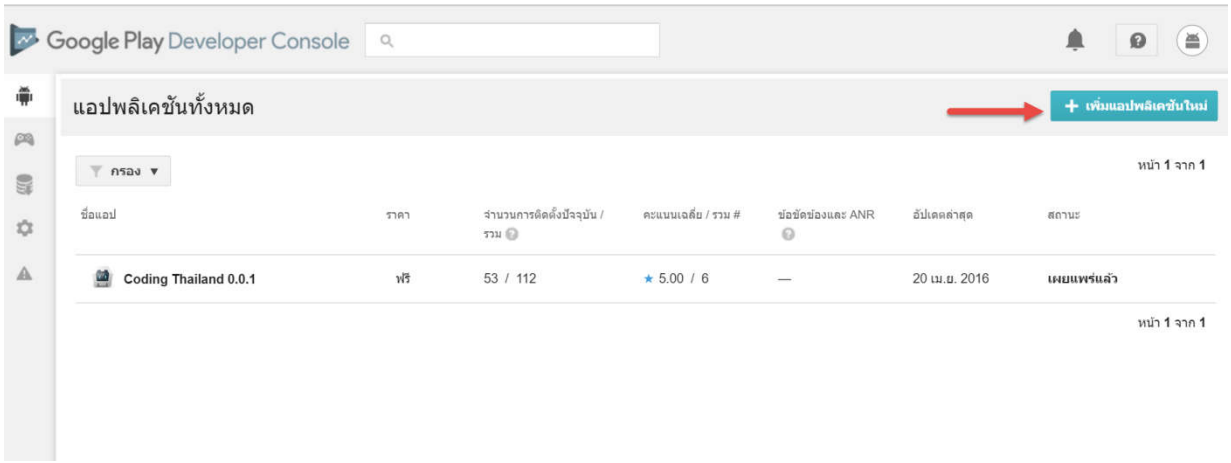
การนำไฟล์ที่ Build เสร็จแล้วอัพโหลดขึ้น Store

ในหัวข้อนี้จะยกตัวอย่างการอัพโหลดไฟล์ .apk ที่เราได้จากการ Build เพื่ออัพโหลดขึ้น Google Play ครับ ขั้นตอนการอัพโหลดมีดังนี้

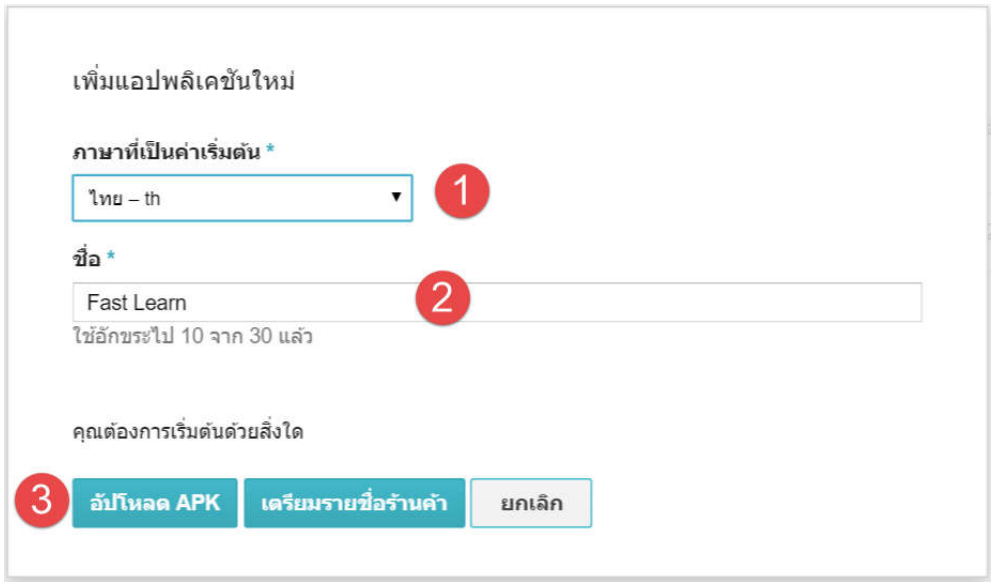
1. เราต้องสมัคร google play developer ก่อนครับ ตามลิงก์นี้ https://play.google.com/apps/publish/signup/ ในขั้นตอนนี้อย่าลืมชำระค่าลงทะเบียนด้วยนะครับ



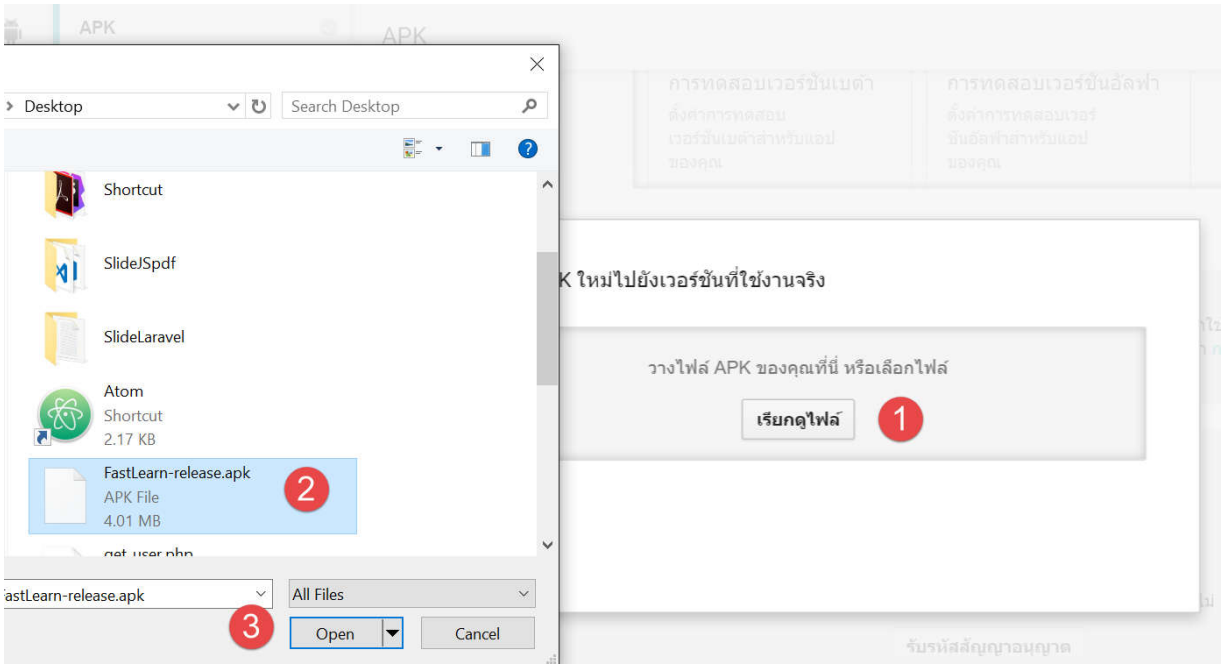
2. เมื่อล็อกอินเข้ามาแล้วให้คลิกปุ่ม เพิ่มแอปพลิเคชันใหม่



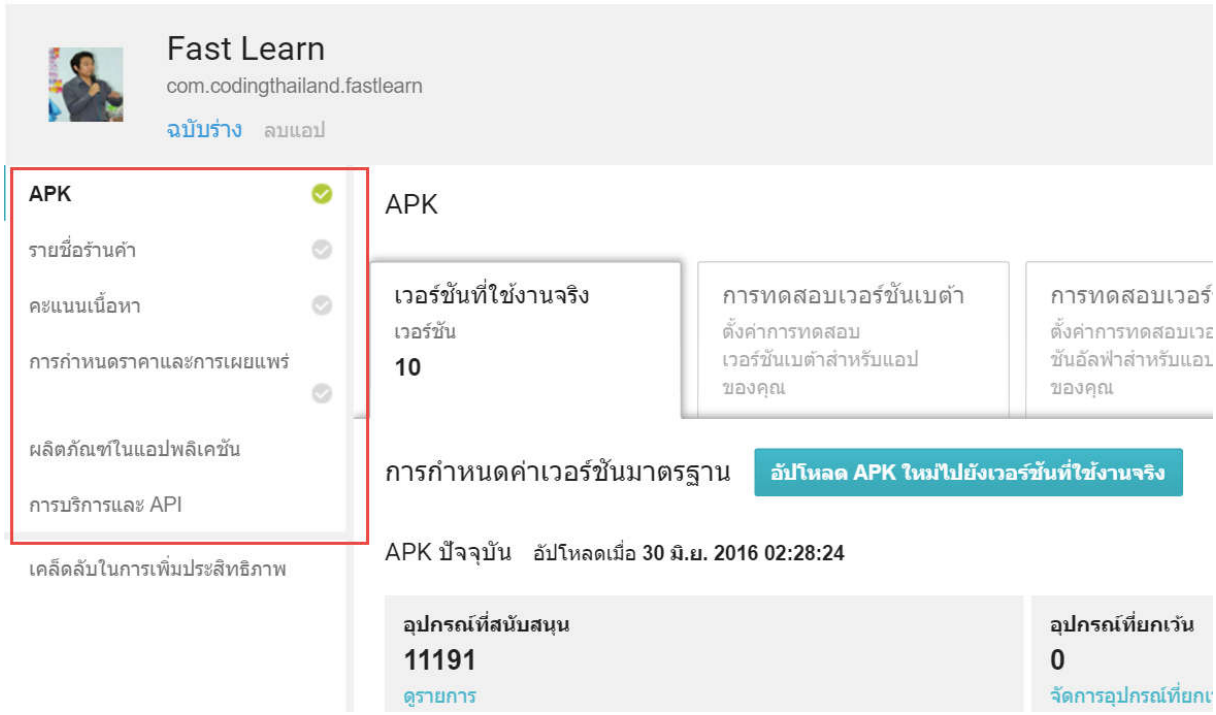
3. เลือกภาษา กรอกชื่อ App และกดอัพโหลด APK



4. คลิกเพื่อเลือกไฟล์ .apk เพื่ออัพโหลด



5. กรอกข้อมูลให้ครบทุกขั้นตอน ตามภาพ



6. หากครบทุกขั้นตอนแล้ว ก็ให้**กดปุ่ม เผยแพร่แอป** ได้เลยครับ App จะสามารถดาวน์โหลดได้ภายใน 1-2 วัน หรือเร็วกว่านั้น

บทที่ 12 โบนัสพิเศษ

- **ตัวอย่างไฟล์การตั้งค่า App เพื่ออัพโหลดไปยัง** PhoneGap Build

  ดาวน์โหลดได้ที่นี่ https://goo.gl/49zunk

- **ตัวอย่างไฟล์** psd (PhotoShop) **สำหรับทำ** splash screen

  ดาวน์โหลดได้ที่นี่ https://goo.gl/jzqz28

- **ไฟล์โค้ดประกอบหนังสือ**

  ดาวน์โหลดได้ที่นี่ https://goo.gl/9Mrbr0

ยินดีด้วยครับ! อ่านถึงหน้าสุดท้ายแล้ว หวังว่าหนังสือเล่มนี้จะช่วยพัฒนาความรู้ และชีวิตทุกคนให้ดีขึ้นนะครับ
หากมีข้อสงสัยตรงไหนก็ถามผมมาได้ตลอดเวลา

ขอบคุณทุกคนครับ

โค้ชเอก

Codingthailand.com