Software Park Thailand

</Code Camp>

SOFTWARE PARK THAILAND

{Digital Academy}

THAI PROGRAMMER

Angular

Forms and Server-side communication

Krissada Chalermsook

SOFTWARE PARK THAILAND

{Digital Academy}

THAI PROGRAMMER

Recap

index.html

```
<!doctype html>
<html lang="en">
<head>
 <meta charset="utf-8">
 <title></title>
 <base href="/">

 <meta name="viewport"
content="width=device-width, initial-
scale=1">
 <link rel="icon" type="image/x-icon"
href="favicon.ico">
</head>
<body>
 <app-root></app-root>
</body>
</html>
```

app.component.ts

```
@Component({
 selector: 'app-root',
 templateUrl: './app.component.html',
 styles: []
})
export class AppComponent implements
OnInit {}
```

app.component.html

```
<div>
 <h2>Show Home Component below</h2>
 <app-home></app-home>
</div>
```

home.component.ts

```
@Component({
 selector: 'app-home',
 templateUrl: './home.component.html',
 styles: []
})
export class HomeComponent implements
OnInit {}
```

home.component.html

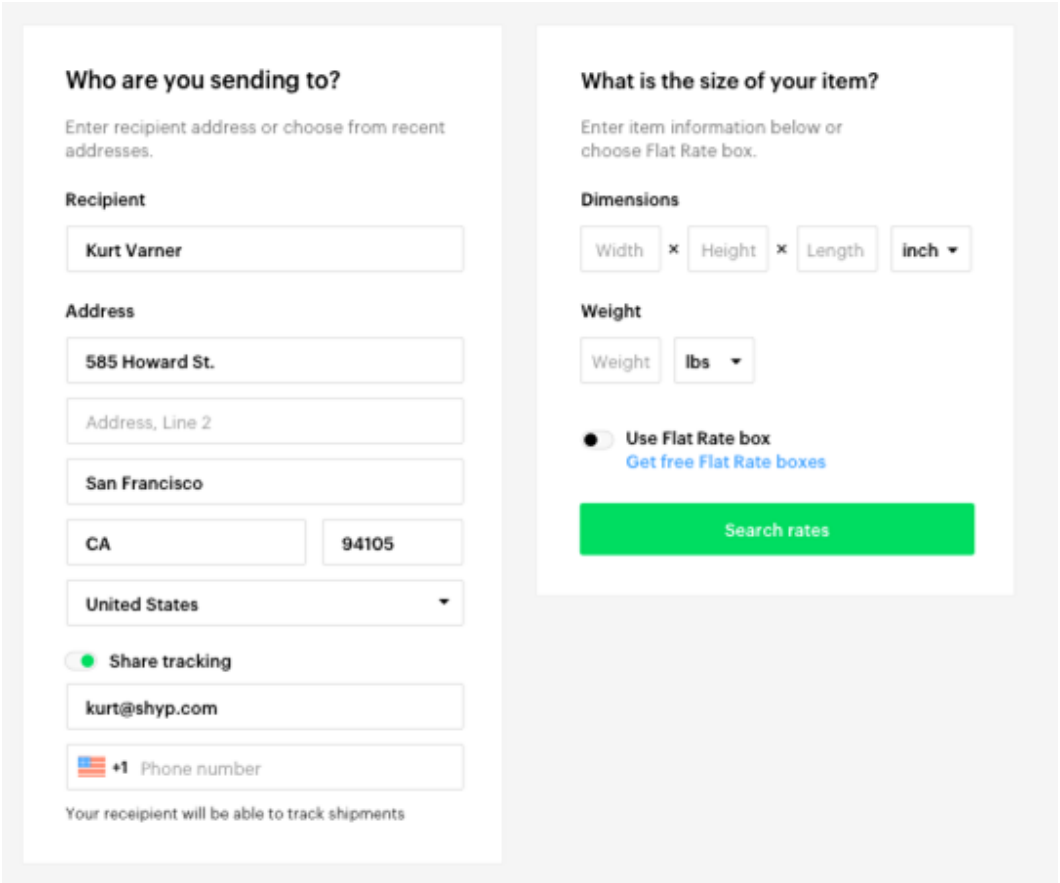
```
<h2>Home Component Work !</h2>
```

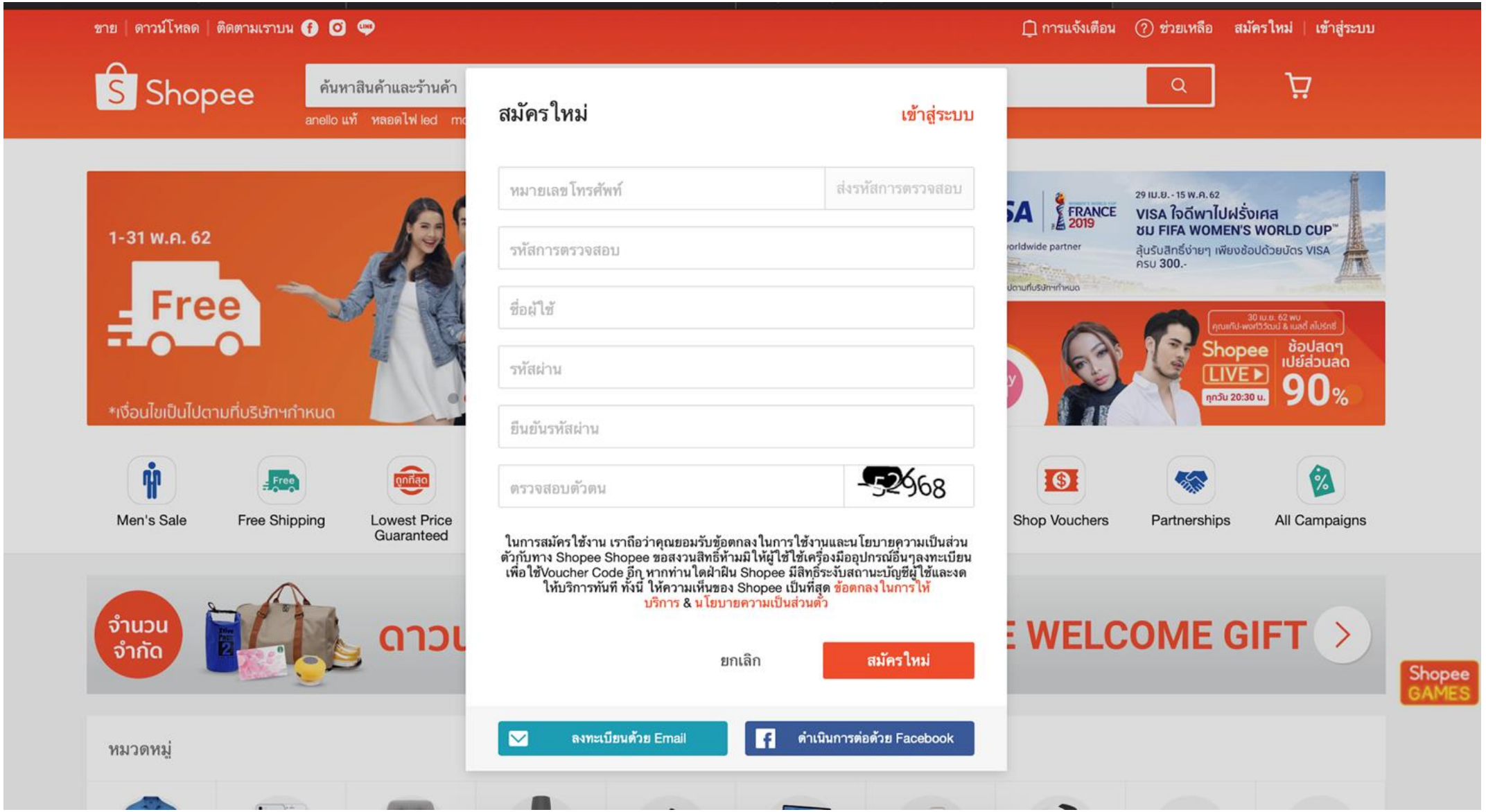
Forms

Software Park Thailand
</Code Camp>

Forms คืออะไร

- คือกลุ่มของ input เพื่อให้ user ป้อนข้อมูลและส่งเข้าสู่ web application หรือส่งไปยัง Backend
- ตัวอย่าง form เช่น form สมัครสมาชิก, ล็อคอินเข้าระบบ
- โดยทั่วไป form จะต้องมีการตรวจสอบความถูกต้องของข้อมูลที่ได้รับเข้ามาก่อนส่งเข้าสู่ backend (ต้องไม่รับข้อมูลขยะ หรือ junk)




ขาย | ดาวน์โหลด | ติดตามเราบน
การแจ้งเตือน
ช่วยเหลือ
สมัครใหม่ | เข้าสู่ระบบ
Shopee
ค้นหาสินค้าและร้านค้า
สมัครใหม่
เข้าสู่ระบบ
หมายเลขโทรศัพท์
ส่งรหัสการตรวจสอบ
รหัสการตรวจสอบ
ชื่อผู้ใช้
รหัสผ่าน
ยืนยันรหัสผ่าน
ตรวจสอบตัวตน
ในการสมัครใช้งาน เราถือว่าคุณยอมรับข้อตกลงในการใช้งานและนโยบายความเป็นส่วนตัวกับทาง Shopee Shopee ขอสงวนสิทธิ์ห้ามมิให้ผู้ใช้ใช้เครื่องมืออุปกรณ์อื่นๆลงทะเบียนเพื่อใช้Voucher Code อีก หากท่านใดฝ่าฝืน Shopee มีสิทธิ์ระงับสถานะบัญชีผู้ใช้และงดให้บริการทันที ทั้งนี้ ให้ความเห็นของ Shopee เป็นที่สุด ข้อตกลงในการให้บริการ & นโยบายความเป็นส่วนตัว
ยกเลิก
สมัครใหม่
ลงทะเบียนด้วย Email
ดำเนินการต่อด้วย Facebook
1-31 พ.ค. 62
Free
*เงื่อนไขเป็นไปตามที่บริษัทฯกำหนด
29 เม.ย. - 15 พ.ค. 62
VISA ใจดีพาไปฝรั่งเศส
ชม FIFA WOMEN'S WORLD CUP
ลุ้นรับสิทธิ์ง่ายๆ เพียงช้อปด้วยบัตร VISA ครบ 300.-
Shopee LIVE
ช้อปสดๆ เปย์ส่วนลด 90%
Men's Sale
Free Shipping
Lowest Price Guaranteed
Shop Vouchers
Partnerships
All Campaigns
จำนวนจำกัด
WELCOME GIFT
Shopee GAMES
หมวดหมู่

Software Park Thailand
</Code Camp>

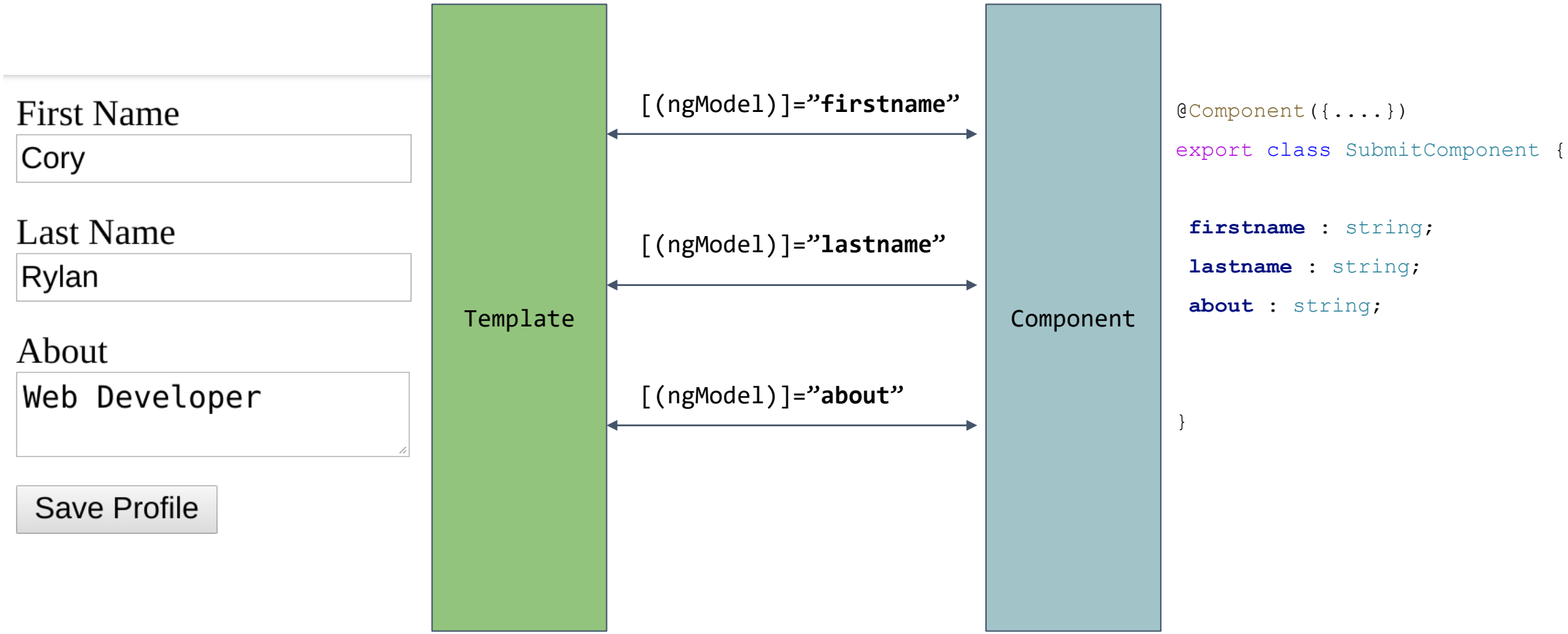
Forms ใน Angular

- มี 2 แบบคือ
  1. Template Driven Form (**FormsModule**)
  2. Reactive Form (**ReactiveFormsModule**)

- จะต้อง import FormsModule, ReactiveFormsModule เข้า Angular module เช่น AppModule
- ทั้ง 2 module import ได้จาก @angular/forms ผ่านคำสั่ง

```
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
```

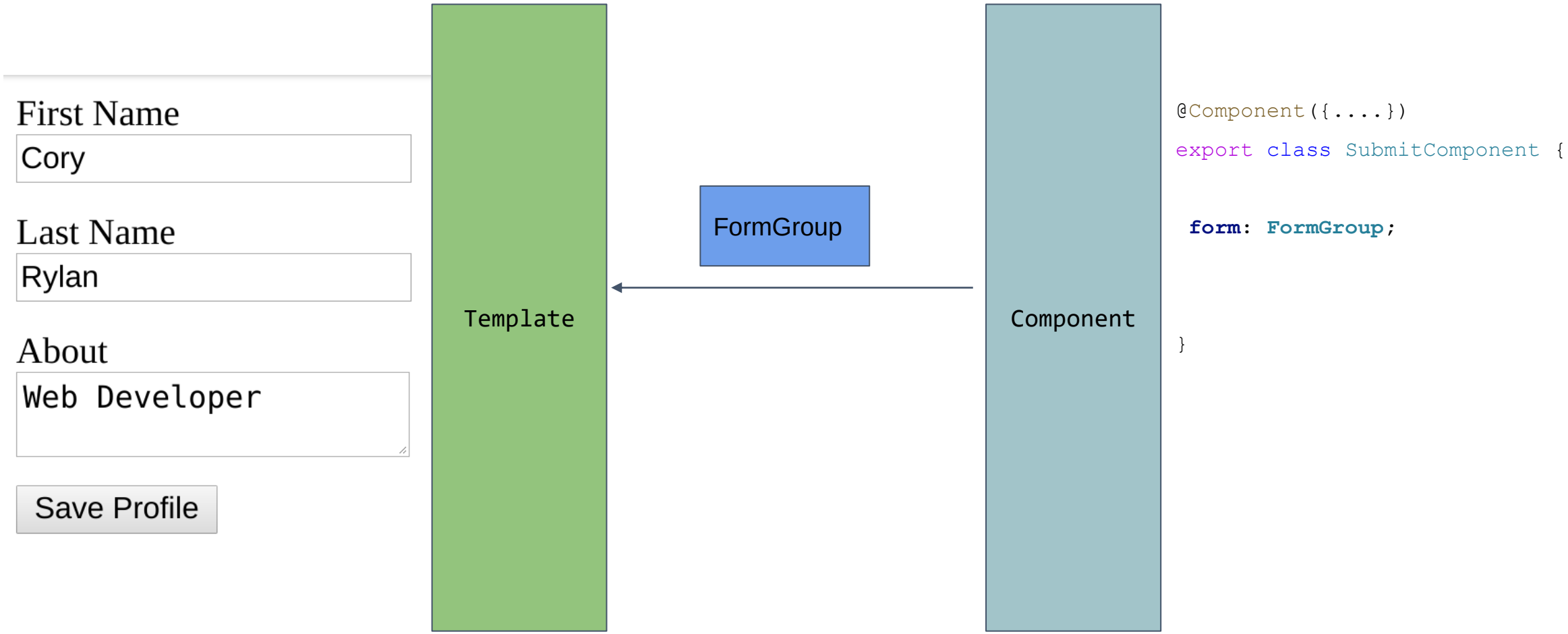
Template Driven Forms

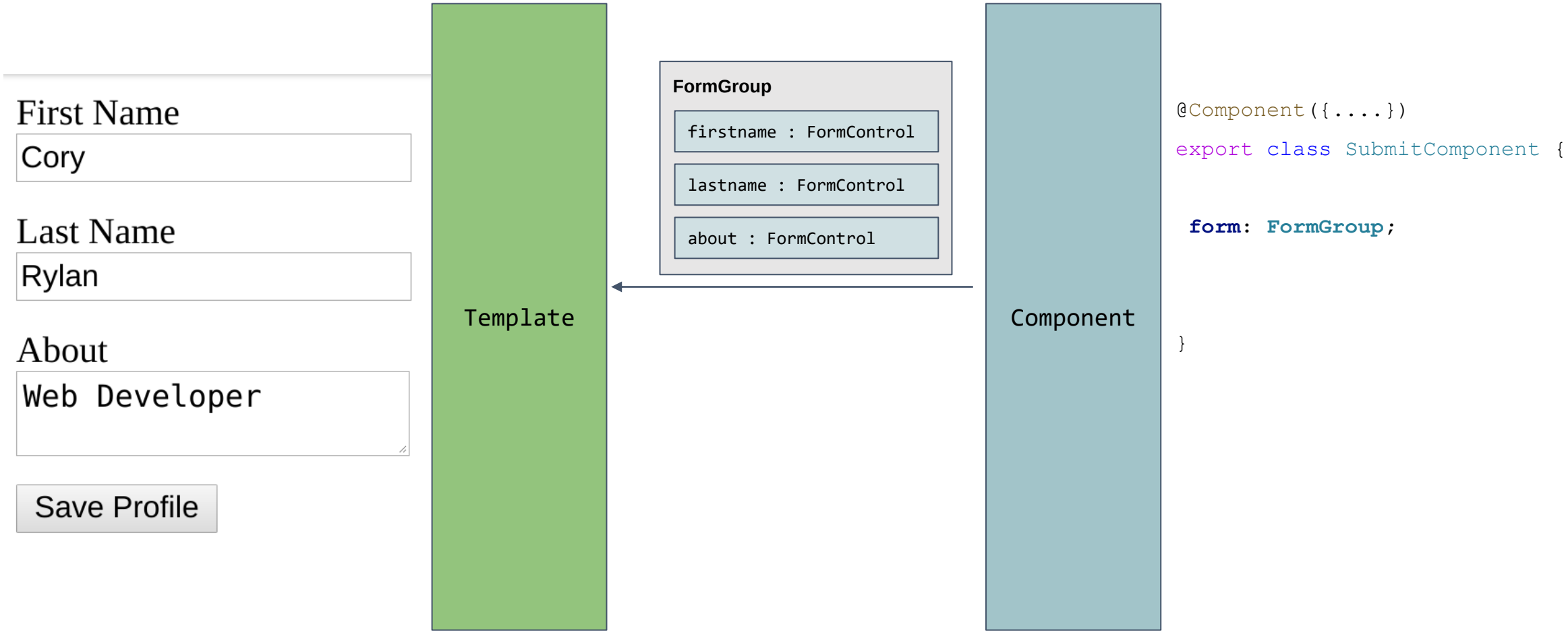

First Name
Cory
Last Name
Rylan
About
Web Developer
Save Profile
Template
[(ngModel)]="firstname"
[(ngModel)]="lastname"
[(ngModel)]="about"
Component
@Component({....})
export class SubmitComponent {
firstname : string;
lastname : string;
about : string;
}

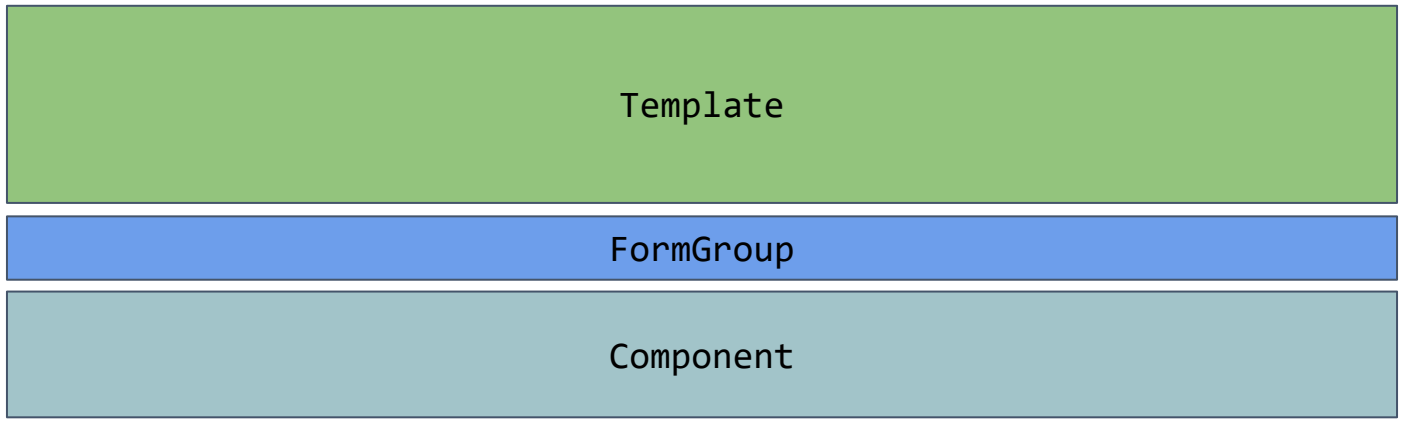
```
<div>
    <div>
        <div>First Name</div>
        <input [(ngModel)]="firstname">
    </div>
    <div>
        <div>Last Name</div>
        <input [(ngModel)]="lastname">
    </div>
    <div>
        <div>About</div>
        <input [(ngModel)]="about">
    </div>
    <div>
        <button>Save Profile</button>
    </div>
</div>
```

ใช้ **[(ngModel)]=”target property name ”** ทำ two-way binding

Reactive Forms


First Name
Cory
Last Name
Rylan
About
Web Developer
Save Profile
Template
FormGroup
Component
@Component({....})
export class SubmitComponent {
form: FormGroup;
}


First Name
Cory
Last Name
Rylan
About
Web Developer
Save Profile
Template
FormGroup
firstname : FormControl
lastname : FormControl
about : FormControl
Component
@Component({....})
export class SubmitComponent {
form: FormGroup;
}


Template
FormGroup
Component

การสร้าง FormGroup บน Component ท่าแรก

```
import { Component, OnInit } from '@angular/core';
import { FormGroup, FormControl } from '@angular/forms';

@Component({
 selector: 'app-submit',
 templateUrl: './submit.component.html',
 styleUrls: ['./submit.component.css']
})
export class SubmitComponent implements OnInit {
 form: FormGroup;
 constructor() {
   this.form = new FormGroup({
     firstname: new FormControl(),
     lastname: new FormControl(),
     about: new FormControl()
   });
 }
 ngOnInit() {}
}
```

1

2

1. สร้าง property ให้เป็นตัวแปรชนิด FormGroup
2. **new instance ที่ constructor** โดยภายในให้ระบุ property ต่างๆ ภายใน form เช่น ใน form มีการรับข้อมูล firstname, lastname และ about โดย property ดังกล่าวเป็นตัวแปรชนิด FormControl

การสร้าง FormGroup บน Component ท่าที่สอง

```
import { Component, OnInit } from '@angular/core';
import { FormGroup, FormControl, FormBuilder } from '@angular/forms';

@Component({
 selector: 'app-submit',
 templateUrl: './submit.component.html',
 styleUrls: ['./submit.component.css']
})
export class SubmitComponent implements OnInit {
 form: FormGroup;
 constructor(private fb: FormBuilder) {
   this.form = this.fb.group({
     firstname: '',
     lastname: '',
     about: ''
   });
 }
 ngOnInit() {}
}
```

1

2

1. สร้าง property ให้เป็นตัวแปรชนิด **FormGroup**
2. **inject FormBuild มาใช้งาน** โดยเรียก method **.group** จะทำให้ภายใน เมื่อระบุ ในแต่ละ FormControl ไม่จำเป็นตัวพิมพ์ keyword คำว่า FormControl (แบบลดรูป)

แบบเต็ม

```
this.form = new FormGroup({
  firstname: new FormControl(),
  lastname: new FormControl(),
  about: new FormControl()
});
```

หรือ

แบบลดรูป (ทำผ่าน FormBuilder)

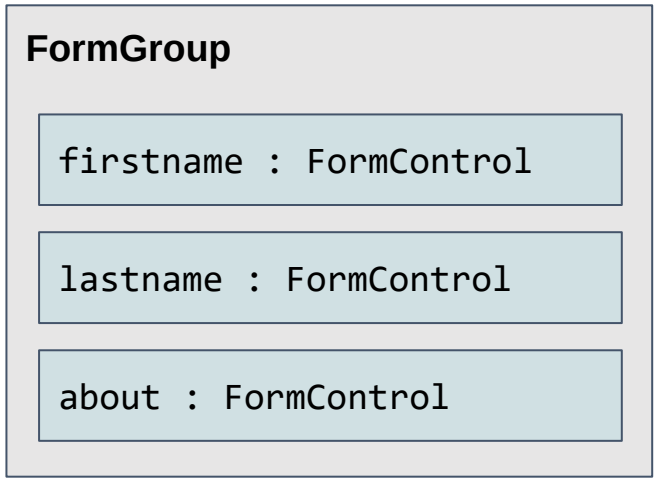
```
this.form = this.fb.group({
    firstname: '',
    lastname: '',
    about: ''
  });
```

แนะนำแบบนี้

Software Park Thailand
</Code Camp>

FormGroup คืออะไร

- เป็นการประกาศ ตัวแปร Form ที่กำลังจะสร้างบน template โดยตัวแปรดังกล่าวเป็นตัวแทนของ Form จริงๆ (Abstraction)
- ภายใน FormGroup มีได้หลาย FormControl
- FormGroup สามารถ export ข้อมูลออกเป็น json objectได้ทันที ผ่าน property **.value** ทำให้ง่ายต่อการสร้าง json object ไม่ต้องเอาในแต่ละ property มาประกอบกันทีหลัง (ประกอบกันมาตั้งแต่ FormGroup แล้ว)



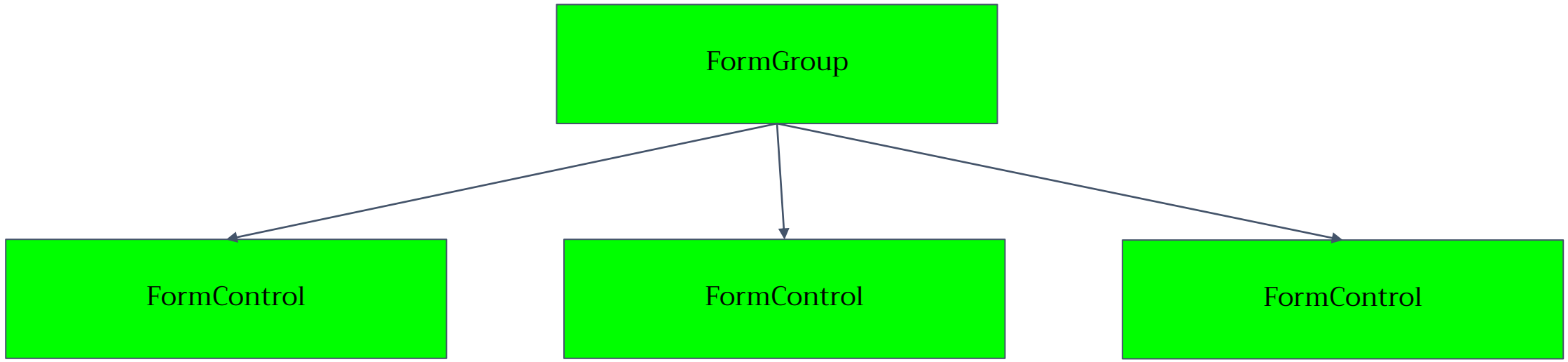
Software Park Thailand
</Code Camp>

FormGroup properties

- **.value** - อ่านค่า FormGroup โดยจะได้ json object (เช่น **this.form.value**)
- **.valid** - อ่านค่า FormGroup ว่า form valid หรือไม่ เงื่อนไขคือ ทุกๆ control ภายใน FormGroup จะต้องไม่มี error (all valid)
- **.invalid** - อ่านค่า FormGroup ว่า form invalid หรือไม่ หากมีแม้แต่ control เดียวภายใน form มี error จะถือว่าทั้ง form invalid
- **.controls** - list จำนวน FormControl ทั้งหมดภายใต้ FormGroup
- .**valueChanges** - สามารถ **subscribe** เพื่อดักจับ change event FormGroup เปลี่ยนแปลงข้อมูลได้

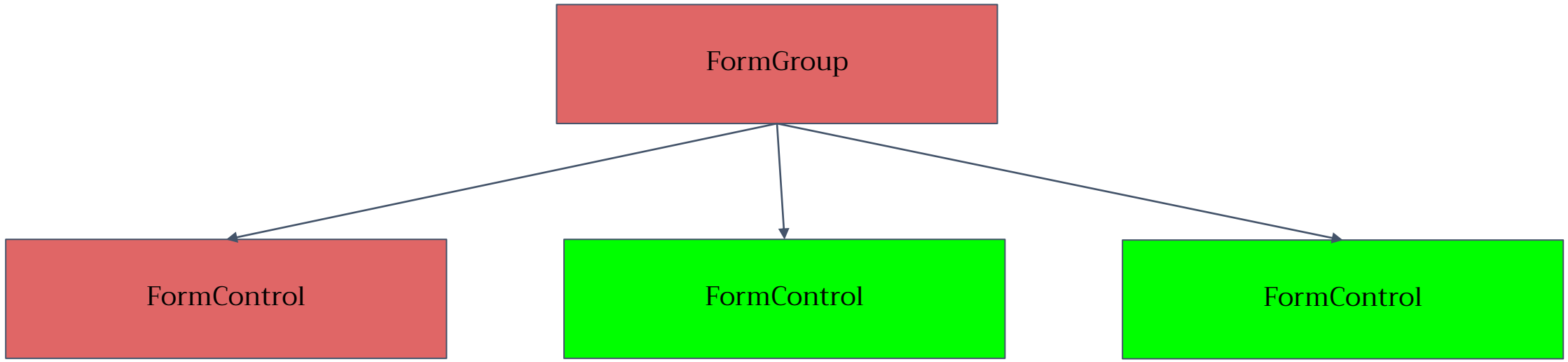
Software Park Thailand
</Code Camp>

Form Valid



Software Park Thailand
</Code Camp>

Form Invalid



Software Park Thailand
</Code Camp>

FormGroup methods

- **.get('<form control name>')** - เรียก FormControl โดยระบุชื่อ ภายใต้ FormGroup เช่น this.form.get('**email**')
- **.reset()** - clear ค่าทุกอย่างที่กรอกบน form
- **.setValue()** - กำหนดค่าให้ทั้งฟอร์ม โดยจะต้องส่ง json object ที่มีจำนวน property ให้ครบตามโครงสร้างเหมือนกับ FormGroup ที่สร้าง เช่น this.form.setValue(**obj**); obj = json object
- **.patchValue()** - กำหนดค่าให้ฟอร์ม โดยส่ง json object ที่มีจำนวน property บางส่วนตามโครงสร้าง กับ FormGroup ที่สร้าง เช่น this.form.**patchValue**(**obj**); obj = json object

การใช้งาน FormGroup บน Template

1

```
<div [formGroup]="form">
    <div>
        <div>First Name</div>
        <input formControlName="firstname">
    </div>
    <div>
        <div>Last Name</div>
        <input formControlName="lastname">
    </div>
    <div>
        <div>About</div>
        <input formControlName="about">
    </div>
    <div>
        <button>Save Profile</button>
    </div>
</div>
```

2

ใช้ attribute directive ชื่อ **[formGroup]** แปะที่ root html element เช่น ตามตัวอย่างมี ครอบ input ต่างๆ แสดงว่า div เป็น root ของ input ต่างๆภายในนั้น **[formGroup] = “ชื่อ property ”**

ใช้ attribute directive ชื่อ **formControlName แปะที่ input** เพื่อให้เชื่อมโยงไปยัง form group ที่ทำใน 1) (ไม่จำเป็นต้องใส่ก้ามปู)

Software Park Thailand
</Code Camp>

Lab 1 - สร้าง Forms โดยใช้ Reactive Forms

1. สร้าง Angular workspace ผ่านคำสั่ง Angular CLI **ng new day4lab1 --interactive false**
2. เปิด workspace folder ใน VSCode และเปิด Angular Dev. Server ด้วยคำสั่ง **ng serve** ใน terminal
3. เปิดอีก terminal (กด + บนขวา)
4. สร้าง MyFormComponent ด้วยคำสั่ง **ng g component my-form --skipTests**
5. เริ่มต้นการสร้าง Reactive Form โดยการเพิ่มโค้ดทีละไฟล์ดังนี้

1

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';
import { MyFormComponent } from './my-form/my-form.component';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';

@NgModule({
 declarations: [AppComponent, MyFormComponent],
 imports: [BrowserModule, FormsModule, ReactiveFormsModule],
 providers: [],
 bootstrap: [AppComponent]
})
export class AppModule {}
```

import **FormsModule และ ReactiveFormsModule** จาก @angular/forms และ import เข้า AppModule

2

my-form.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormGroup, FormBuilder } from '@angular/forms';

@Component({
 selector: 'app-my-form',
 templateUrl: './my-form.component.html',
 styleUrls: ['./my-form.component.css']
})
export class MyFormComponent implements OnInit {
 form: FormGroup;

 constructor(private fb: FormBuilder) {

 }

 ngOnInit() {}
}
```

1

2

1. สร้าง property ให้เป็นตัวแปรชนิด **FormGroup**
2. **inject FormBuild มาใช้งาน**

3

my-form.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormGroup, FormBuilder } from '@angular/forms';

@Component({
 selector: 'app-my-form',
 templateUrl: './my-form.component.html',
 styleUrls: ['./my-form.component.css']
})
export class MyFormComponent implements OnInit {
 form: FormGroup;

 constructor(private fb: FormBuilder) {
   this.form = this.fb.group({
     phoneNo: '',
     email: '',
     password: '',
     confirmPassword: ''
   });
 }
 ngOnInit() {}
}
```

3

3. เรียก method **.group จาก FormBuilder** จะทำให้ภายใน เมื่อระบุ ในแต่ละ FormControl ไม่จำเป็นตัวพิมพ์ keyword คำว่า FormControl (แบบลดรูป)

4

my-form.component.html

```
<form [formGroup]="form">
 <div>
   Phone : <input formControlName="phoneNo">
 </div>
 <div>
   Email : <input formControlName="email">
 </div>
 <div>
   Password : <input type="password" formControlName="password">
 </div>
 <div>
   Confirm Password : <input type="password" formControlName="confirmPassword">
 </div>
</form>
<p>
 Form Data Below
</p>
<code>
 {{form.value | json}}
</code>
```

1

ใช้ attribute directive ชื่อ **[formGroup]** แปะที่ root html element เช่น ตามตัวอย่างมี ครอบ input ต่างๆ แสดงว่า div เป็น root ของ input ต่างๆภายในนั้น
[formGroup] = “ชื่อ property ”

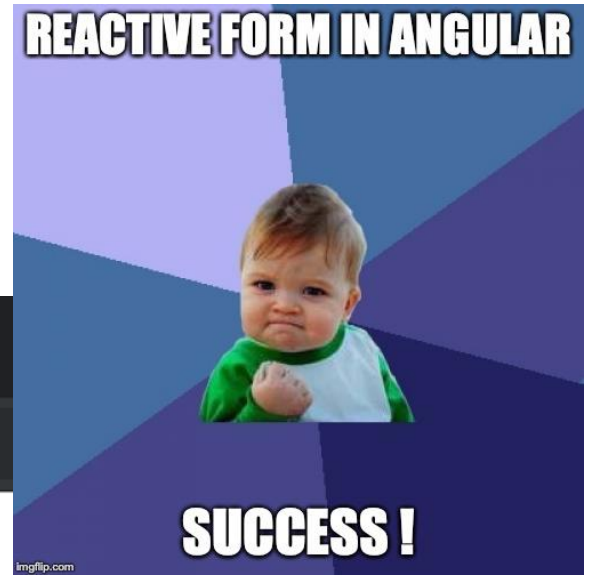
2

ใช้ attribute directive ชื่อ **formControlName แปะที่ input** เพื่อให้เชื่อมโยงไปยัง form group ที่ทำใน 1) (ไม่จำเป็นต้องใส่ก้ามปู)

app.component.html

```
<div>
  <app-my-form></app-my-form>
</div>
```

6



Day31lab1

localhost:4200

Phone : 12345
Email : nitipan.p@codecamp.co
Password : •••••
Confirm Password : •••••

Form Data Below

```
{ "phoneNo": "12345", "email": "nitipan.p@codecamp.com", "password": "admin", "confirmPassword": "admin" }
```

Validations

Software Park Thailand
</Code Camp>

Validators class

- เป็น class ที่ import ได้จาก @angular/forms
- มี set ของ build-in ฟังก์ชันการ validate forms ได้แก่
  - min
  - max
  - email
  - required
  - minlength
  - maxlength
- วิธีเรียกใช้งาน ต้องประกาศขณะสร้าง FormGroup เช่น **Validators.email()**

```
this.form = this.fb.group({
      phoneNo: [''],
      email: ['', Validators.email],
      password: ['', [Validators.required, Validators.minLength(8)]],
      confirmPassword: ['', Validators.required]
    });
```

รูปแบบการใส่ Validators จะต้อง ใส่ในก้ามปู ขณะสร้าง FormGroup ในแต่ละ FormControl ซึ่งสามารถมีได้มาก 1 Validators หากมีมากกว่า 1 ให้ใส่ ก้ามปูครอบอีกที (เพื่อบอกว่าเป็น array)

ใช้ .get() form control ซึ่งเป็น method จาก FormGroup ในการเข้าถึง control นั้นว่ามี errors หรือไม่ หากมี error จะสามารถนำมาเป็นเงื่อนไขบน template ได้

```
 <div>
   Password : <input type="password" formControlName="password">
   <span style="color: red;" *ngIf="form.get('password').errors?.required">Password
required</span>
   <span style="color: red;" *ngIf="form.get('password').errors?.minlength">Password should be at
least 8
     characters</span>
 </div>
```

error name หาก มี error จาก form group จะมี ค่า property name เกิดขึ้น (ดูสไลด์ถัดไป) แต่หาก ไม่มี error (คือ form control นั้น valid) errors จะมีค่าเป็น null

```
form.get(<control name>).errors?.<error name>
```

Tip: ใส่ ? เพื่อให้ตัวแปร nullable สามารถใช้งานบน template ได้โดยไม่มี error หากเกิดค่า null (property errors เป็น nullable)

Software Park Thailand
</Code Camp>

Error name from Build-in validations

- max
- min
- required
- email
- minlength
- maxlength

Software Park Thailand
</Code Camp>

Lab 2: ทดสอบ Validations

1. ทำต่อจาก lab 1 โดยการเพิ่ม validation ไปยัง property **form** ของ MyFormComponent (แก้ไขการสร้าง form จาก FormBuilder)
2. แก้ไข template ของ MyFormComponent
3. ทดสอบ ว่า form validate ถูกต้องหรือไม่โดยเข้าไปทดสอบบน browser

1

my-form.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

@Component({
 selector: 'app-my-form',
 templateUrl: './my-form.component.html',
 styleUrls: ['./my-form.component.css']
})
export class MyFormComponent implements OnInit {
 form: FormGroup;

 constructor(private fb: FormBuilder) {
   this.form = this.fb.group({
     phoneNo: [''],
     email: ['', Validators.email],
     password: ['', [Validators.required, Validators.minLength(8)]],
     confirmPassword: ['', Validators.required]
   });
 }
 ngOnInit() {}
}
```

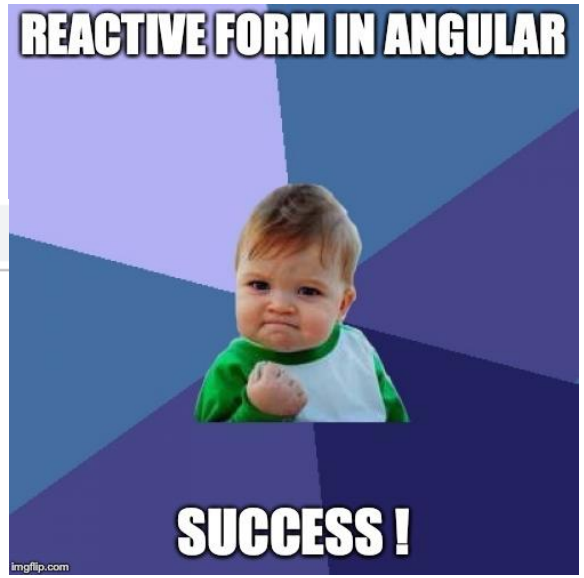
แก้ไข form group ให้ใช้ Validators กำกับในแต่ละ form control

2

my-form.component.html

```
<form [formGroup]="form">
 <div>
   Phone : <input formControlName="phoneNo">
 </div>
 <div>
   Email : <input formControlName="email">
  <span style="color: red;" *ngIf="form.get('email').errors?.email">Email is invalid</span>
 </div>
 <div>
   Password : <input type="password" formControlName="password">
   <span style="color: red;" *ngIf="form.get('password').errors?.required">Password required</span>
   <span style="color: red;" *ngIf="form.get('password').errors?.minlength">Password should be at least 8
     characters</span>

 </div>
 <div>
   Confirm Password : <input type="password" formControlName="confirmPassword">
   <span style="color: red;" *ngIf="form.get('confirmPassword').errors?.required">Confirm Password required</span>
 </div>
</form>
<p>
 Form Data Below
</p>
<code>
 {{form.value | json}}
</code>
```

3



localhost:4200

Phone : 123

Email : aaa@AAAA- Email is invalid

Password : •••• Password should be at least 8 characters

Confirm Password : Confirm Password required

Form Data Below

```
{ "phoneNo": "123", "email": "aaa@AAAA-", "password": "1234", "confirmPassword": "" }
```

มีต่อ !!

4

my-form.component.html

```
<form [formGroup]="form">
 <div>
   Phone : <input formControlName="phoneNo">
 </div>
 <div>
   Email : <input formControlName="email">
  <span style="color: red;" *ngIf="form.get('email').errors?.email">Email is invalid</span>
 </div>
 <div>
   Password : <input type="password" formControlName="password">
   <span style="color: red;" *ngIf="form.get('password').errors?.required">Password required</span>
   <span style="color: red;" *ngIf="form.get('password').errors?.minlength">Password should be at least 8
     characters</span>

 </div>
 <div>
   Confirm Password : <input type="password" formControlName="confirmPassword">
   <span style="color: red;" *ngIf="form.get('confirmPassword').errors?.required">Confirm Password required</span>
 </div>
<button [disabled]="form.invalid" (click)="submitForm()">Submit</button>
</form>
<p>
 Form Data Below
</p>
<code>
 {{form.value | json}}
</code>
```

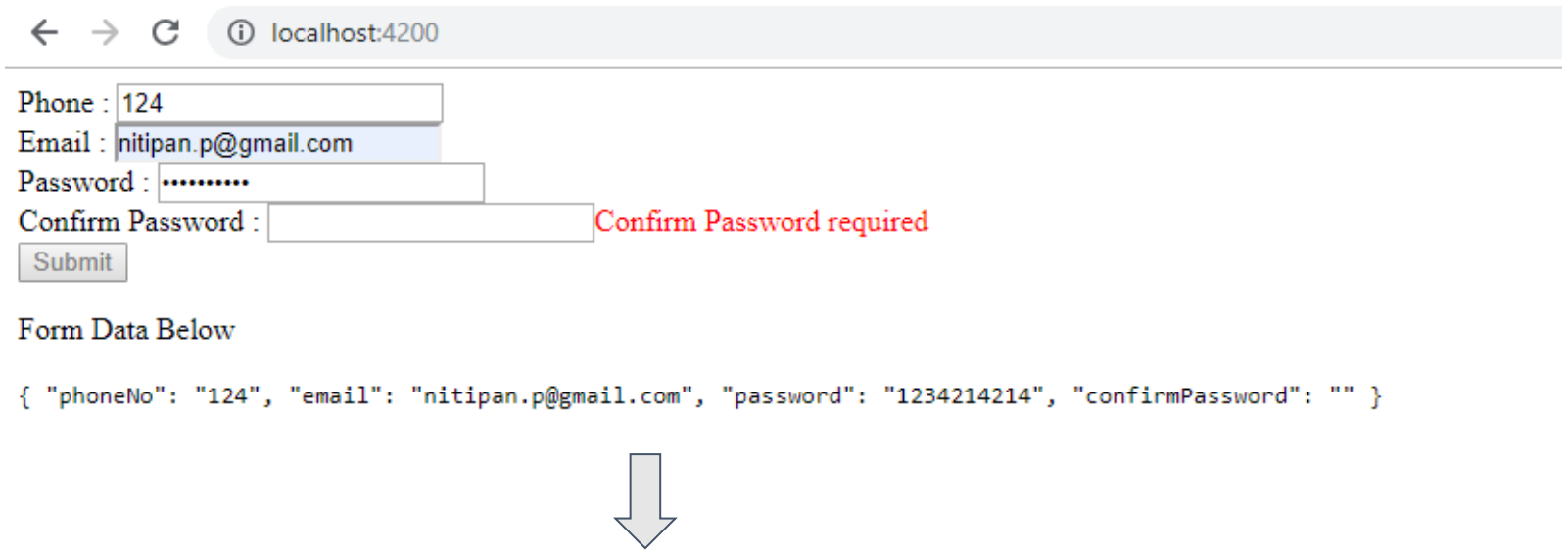
เพิ่มปุ่ม และเงื่อนไข ปุ่มจะ disabled หาก form.invalid เป็น true (แสดงว่าทุก control ต้อง valid)

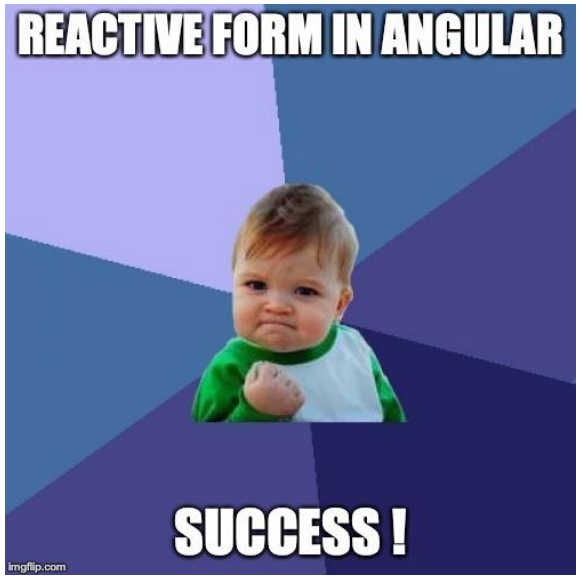
5

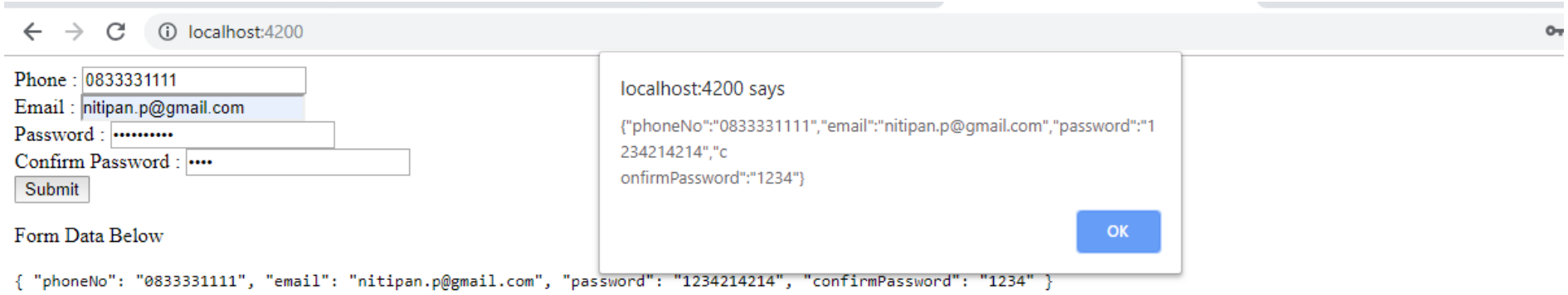
my-form.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

@Component({
 selector: 'app-my-form',
 templateUrl: './my-form.component.html',
 styleUrls: ['./my-form.component.css']
})
export class MyFormComponent implements OnInit {
 form: FormGroup;

 constructor(private fb: FormBuilder) {
   this.form = this.fb.group({
     phoneNo: [''],
     email: ['', Validators.email],
     password: ['', [Validators.required, Validators.minLength(8)]],
     confirmPassword: ['', Validators.required]
   });
 }
 ngOnInit() {}

 submitForm() {
   alert(JSON.stringify(this.form.value));
 }
}
```

6


localhost:4200
Phone : 124
Email : nitipan.p@gmail.com
Password : ••••••••••
Confirm Password :
Confirm Password required
Submit
Form Data Below
{ "phoneNo": "124", "email": "nitipan.p@gmail.com", "password": "1234214214", "confirmPassword": "" }


REACTIVE FORM IN ANGULAR
SUCCESS !
imgflip.com

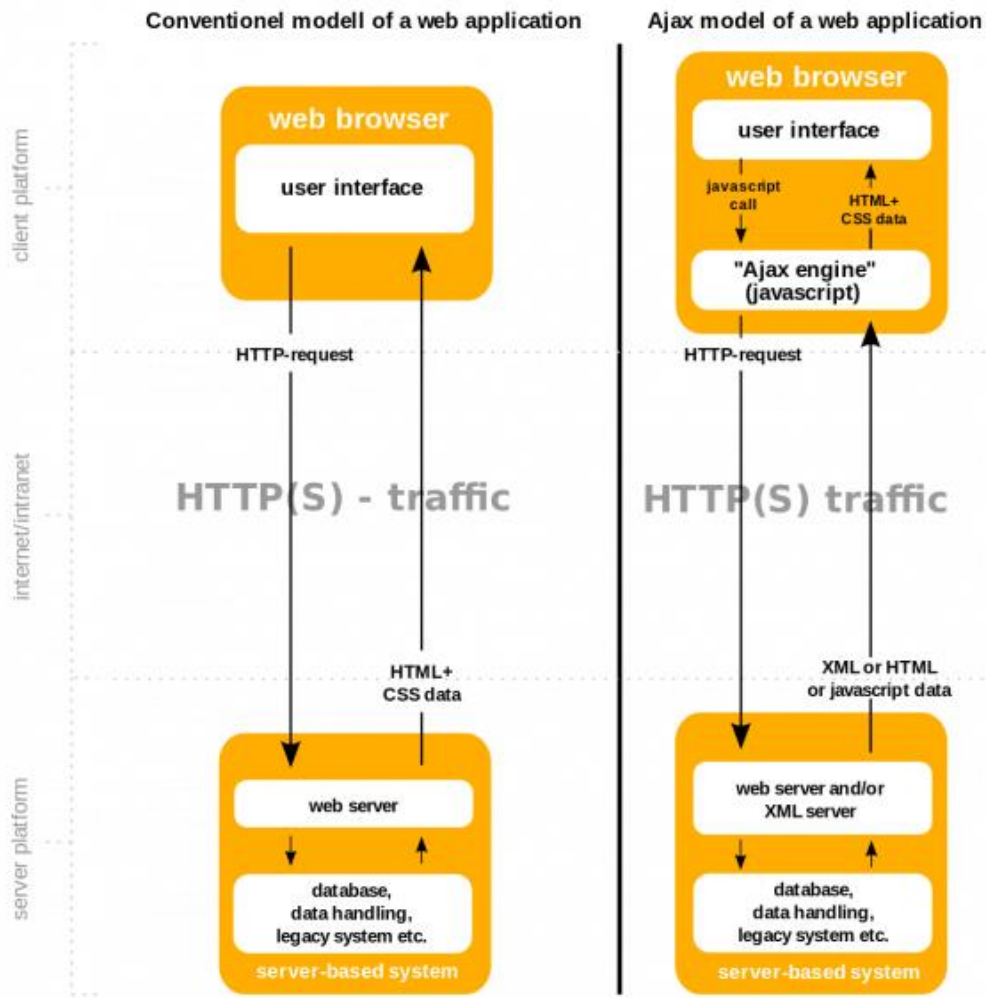

localhost:4200
Phone : 0833331111
Email : nitipan.p@gmail.com
Password : ••••••••••
Confirm Password : ••••
Submit
Form Data Below
{ "phoneNo": "0833331111", "email": "nitipan.p@gmail.com", "password": "1234214214", "confirmPassword": "1234" }
localhost:4200 says
{"phoneNo":"0833331111","email":"nitipan.p@gmail.com","password":"1234214214","confirmPassword":"1234"}
OK

Server-side communication

Software Park Thailand
</Code Camp>

Server-side communication

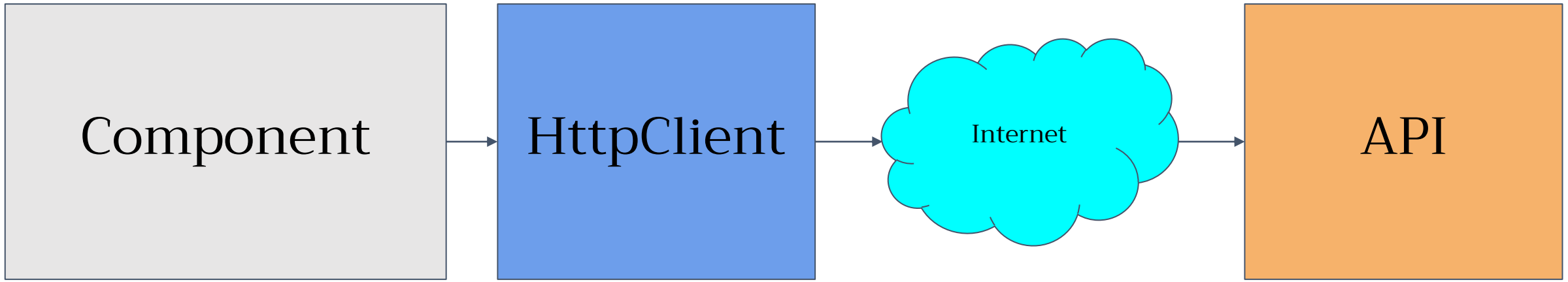
- ใช้เทคนิคชื่อ AJAX (Asynchronous JavaScript XML)
- AJAX จะใช้ object ชื่อ XMLHttpRequest (XHR) ที่ถูก implement โดย browser ในการสื่อสารกับ server
- การใช้ AJAX สื่อสารกับ server ทำให้ไม่จำเป็นต้องโหลดใหม่ทั้งหน้า
- โดยทั่วไปการสื่อสารแบบ AJAX จะใช้ข้อมูลชนิด JSON รับ-ส่งระหว่าง client และ server
- ทำได้ทั้ง GET,POST, PUT, DELETE เป็นต้น
- ใน Angular การสื่อสารด้วย AJAX จะต้อง import **HttpClientModule**


Conventionel modell of a web application
Ajax model of a web application
client platform
internet/intranet
server platform
web browser
user interface
HTTP-request
HTTP(S) - traffic
HTML+
CSS data
web server
database,
data handling,
legacy system etc.
server-based system
web browser
user interface
javascript
call
HTML+
CSS data
"Ajax engine"
(javascript)
HTTP-request
HTTP(S) traffic
XML or HTML
or javascript data
web server and/or
XML server
database,
data handling,
legacy system etc.
server-based system

Software Park Thailand
</Code Camp>

HttpClientModule

- เมื่อ import **HttpClientModule** มาใช้งานจะสามารถนำ **HttpClient** ไป inject เข้า component เพื่อใช้งาน AJAX ได้
- การจะเข้าถึงข้อมูลเมื่อใช้ HttpClient เชื่อมต่อจะต้อง **subscribe** เอา JSON


Component
HttpClient
Internet
API

Software Park Thailand
</Code Camp>

การ import HttpClientModule

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';
import { HttpClientModule } from '@angular/common/http';

@NgModule({
 declarations: [AppComponent],
 imports: [BrowserModule, HttpClientModule],
 providers: [],
 bootstrap: [AppComponent]
})
export class AppModule {}
```

จะต้อง import **HttpClientModule** หากจะใช้ class HttpClient

Software Park Thailand
</Code Camp>

การ inject HttpClient เพื่อใช้งาน AJAX

```
import { Component, OnInit } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Component({
 selector: 'app-repo-list',
 templateUrl: './repo-list.component.html',
 styleUrls: ['./repo-list.component.css']
})
export class RepoListComponent implements OnInit {
 constructor(private httpClient: HttpClient) {}

 ngOnInit() {}
}
```

inject HttpClient ไปที่ constructor ของ component ที่ต้องการใช้งาน Ajax

Software Park Thailand
</Code Camp>

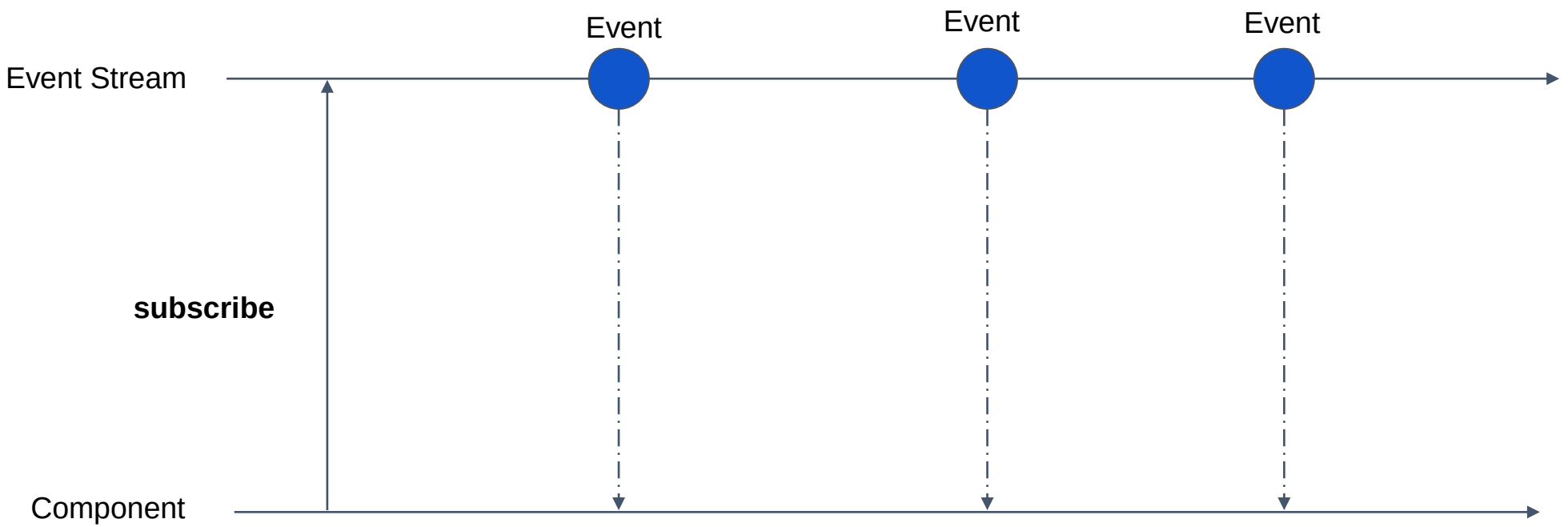
การเรียกเรียกใช้ HttpClient

```
import { Component, OnInit } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Component({
 selector: 'app-repo-list',
 templateUrl: './repo-list.component.html',
 styleUrls: ['./repo-list.component.css']
})
export class RepoListComponent implements OnInit {
 data: any[];
 constructor(private httpClient: HttpClient) {}

 ngOnInit() {
   this.httpClient.get('<url api>').subscribe(result => {
     ///
   });
 }
}
```

เพื่อเรียกฟังก์ชัน โดยอ้างอิงกับ http method ไม่ว่าจะเป็น get หรือ post เป็นต้นให้ **subscribe** เอาข้อมูลมาใช้งาน ข้อมูลดังกล่าวคือสิ่งที่ได้กลับมาจาก server

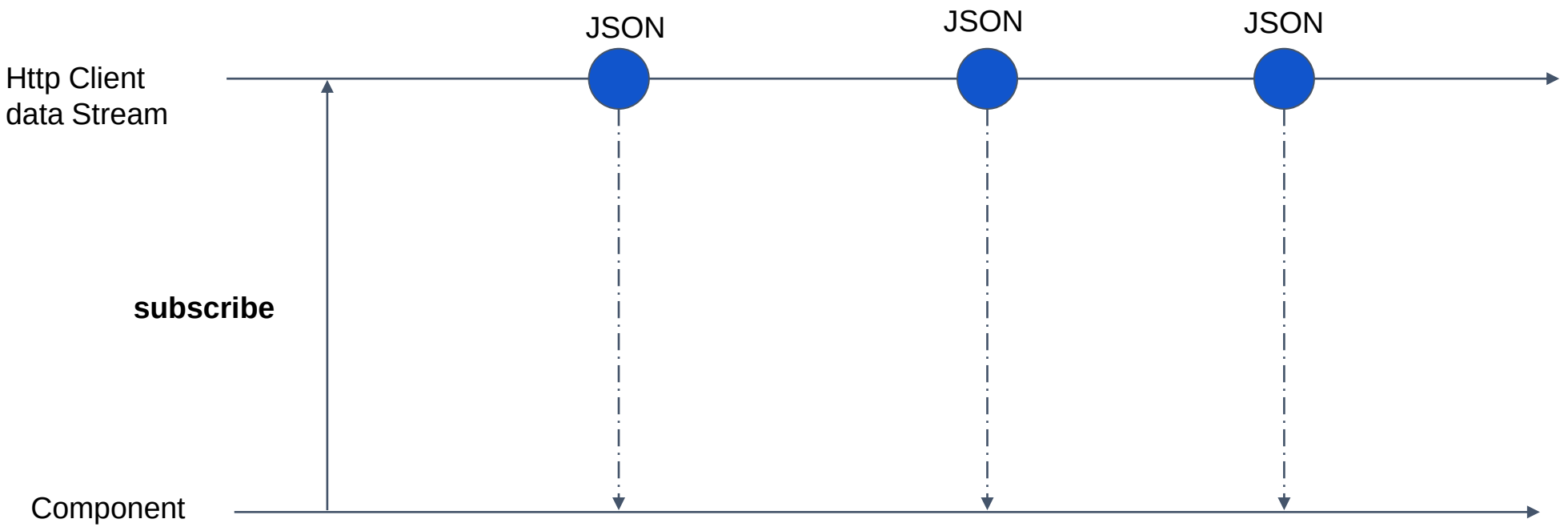
Software Park Thailand
</Code Camp>

Reactive Programming 101



Software Park Thailand
</Code Camp>

Reactive Programming 101



Software Park Thailand
</Code Camp>

แนวทางการเตรียมการก่อนที่จะทำการเชื่อมต่อไปที่ API

1. ต้องรู้ก่อนว่า structure ของข้อมูล ที่จะต่อไปที่ API มีลักษณะอย่างไร เช่น return กลับมาเป็น JSON หรือ array ของ JSON
2. ลองนำ **Postman** มายิงดูข้อมูลก่อน เพราะ Tool ดังกล่าวจะแสดงผลให้เข้าใจง่าย (มีการ format JSON)
3. วางแผนการนำข้อมูลไปแสดงบน template ของ component

Software Park Thailand
</Code Camp>

Example API

- https://jsonplaceholder.typicode.com/
- มี API ให้ลองเล่น

/posts	100 posts
/comments	500 comments
/albums	100 albums
/photos	5000 photos
/todos	200 todos
/users	10 users

Software Park Thailand
</Code Camp>

Example API

jsonplaceholder.typicode.com/posts


```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
    "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum e
  },
  {
    "userId": 1,
    "id": 2,
    "title": "qui est esse",
    "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditiis voluptate porro vel
debitis possimus qui neque nisi nulla"
  },
  {
    "userId": 1,
    "id": 3,
    "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut",
    "body": "et iusto sed quo iure\nvoluptatem occaecati omnis eligendi aut ad\nvoluptatem doloribus vel accusantium quis pariatur\nmolest
  },
  {
    "userId": 1,
    "id": 4,
    "title": "eum et est occaecati",
    "body": "ullam et saepe reiciendis voluptatem adipisci\nsit amet autem assumenda provident rerum culpa\nquis hic commodi nesciunt rem
voluptatem rerum illo velit"
  },
  {
    "userId": 1,
    "id": 5,
    "title": "nesciunt quas odio",
    "body": "repudiandae veniam quaerat sunt sed\nalias aut fugiat sit autem sed est\nvoluptatem omnis possimus esse voluptatibus quis\nnes
  },
```

Software Park Thailand
</Code Camp>

Lab 3 : ทดสอบ AJAX โดยใช้ HttpClient

1. สร้าง Angular workspace ผ่านคำสั่ง Angular CLI **ng new day4lab3 --interactive false**
2. เปิด workspace folder ใน VSCode และเปิด Angular Dev. Server ด้วยคำสั่ง **ng serve** ใน terminal
3. เปิดอีก terminal (กด + บนขวา)
4. สร้าง PostListComponent ด้วยคำสั่ง **ng g component post-list --skipTests**
5. เริ่มต้นการทดสอบ HttpClient เพื่อใช้ AJAX ต่อไปที่ server โดยการเพิ่มโค้ดทีละไฟล์ดังนี้

1

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';
import { HttpClientModule } from '@angular/common/http';
import { PostListComponent } from './post-list/post-list.component';

@NgModule({
 declarations: [AppComponent, PostListComponent],
 imports: [BrowserModule, HttpClientModule],
 providers: [],
 bootstrap: [AppComponent]
})
export class AppModule {}
```

จะต้อง import **HttpClientModule** หากจะใช้ class HttpClient

2

post-list.component.ts

```
import { Component, OnInit } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Component({
 selector: 'app-post-list',
 templateUrl: './post-list.component.html',
 styleUrls: ['./post-list.component.css']
})
export class PostListComponent implements OnInit {
 constructor(private httpClient: HttpClient) {}

 ngOnInit() {}
}
```

inject HttpClient เข้า PostListComponent เพื่อให้ใช้งาน HttpClient ได้

3

post-list.component.ts

```
import { Component, OnInit } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Component({
 selector: 'app-post-list',
 templateUrl: './post-list.component.html',
 styleUrls: ['./post-list.component.css']
})
export class PostListComponent implements OnInit {
 posts: any[];

 constructor(private httpClient: HttpClient) {}

 ngOnInit() {
   this.httpClient
     .get('https://jsonplaceholder.typicode.com/posts')
     .subscribe(result => {
       this.posts = result as any[];
     });
 }
}
```

Note: หากใช้ relative path เช่น ระบุแค่ api/posts จะต้อง config ให้ Angular ใช้ Proxy (https://angular.io/guide/build#proxying-to-a-backend-server) ไม่เช่นนั้น Ajax จะพยาม request ไปที่ port 4200 ซึ่งไม่มี API นี้อยู่จริง

4

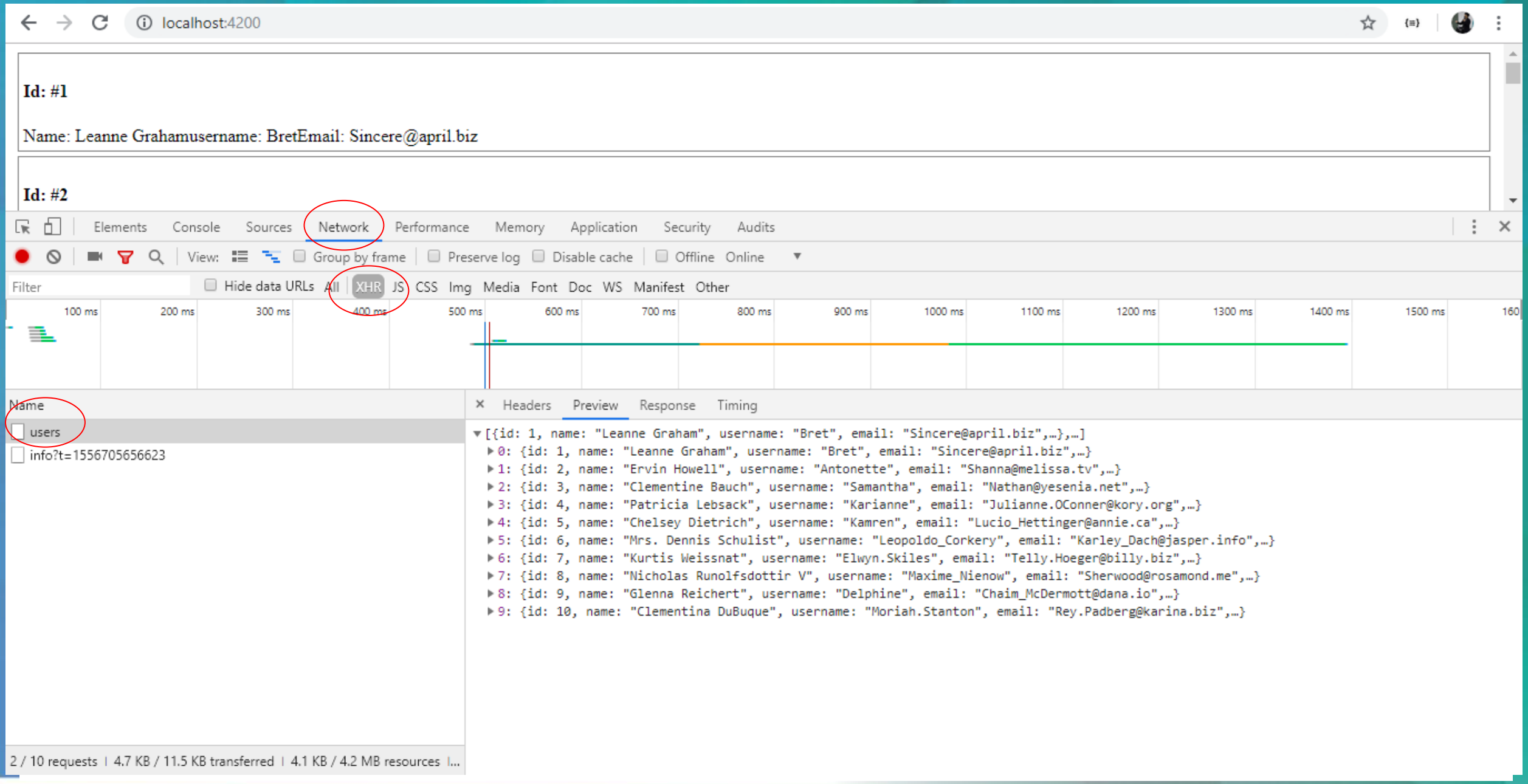
post-list.component.html

```
<div *ngFor="let post of posts"
style="border:1px solid
grey;margin:4px;padding:4px;">
 <h4>User Id: #{{post.userId}}</h4>
 <span>Id: {{post.id}}</span>
 <span>Title: {{post.title}}</span>
 <span>Body: {{post.body}}</span>
</div>
```


```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaeca
    "body": "quia et suscipit\nsuscipit recusandae conse
  },
  {
    "userId": 1,
    "id": 2,
    "title": "qui est esse",
    "body": "est rerum tempore vitae\nsequi sint nihil r
debitis possimus qui neque nisi nulla"
  },
  {
    "userId": 1,
    "id": 3,
    "title": "ea molestias quasi exercitationem repellat
    "body": "et iusto sed quo iure\nvoluptatem occaecati
  },
  {
    "userId": 1,
    "id": 4,
    "title": "eum et est occaecati",
    "body": "ullam et saepe reiciendis voluptatem adipis
voluptatem rerum illo velit"
  },
  {
    "userId": 1,
    "id": 5,
    "title": "nesciunt quas odio",
    "body": "repudiandae veniam quaerat sunt sed\nalias
  },
```

app.component.html

```
<div>
   <app-post-list></app-post-list>
</div>
```


localhost:4200
Id: #1
Name: Leanne Grahamusername: BretEmail: Sincere@april.biz
Id: #2
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
View:
Group by frame
Preserve log
Disable cache
Offline
Online
Filter
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Manifest
Other
100 ms
200 ms
300 ms
400 ms
500 ms
600 ms
700 ms
800 ms
900 ms
1000 ms
1100 ms
1200 ms
1300 ms
1400 ms
1500 ms
Name
users
info?t=1556705656623
Headers
Preview
Response
Timing
[{id: 1, name: "Leanne Graham", username: "Bret", email: "Sincere@april.biz",…},…]
0: {id: 1, name: "Leanne Graham", username: "Bret", email: "Sincere@april.biz",…}
1: {id: 2, name: "Ervin Howell", username: "Antonette", email: "Shanna@melissa.tv",…}
2: {id: 3, name: "Clementine Bauch", username: "Samantha", email: "Nathan@yesenia.net",…}
3: {id: 4, name: "Patricia Lebsack", username: "Karianne", email: "Julianne.OConner@kory.org",…}
4: {id: 5, name: "Chelsey Dietrich", username: "Kamren", email: "Lucio_Hettinger@annie.ca",…}
5: {id: 6, name: "Mrs. Dennis Schulist", username: "Leopoldo_Corkery", email: "Karley_Dach@jasper.info",…}
6: {id: 7, name: "Kurtis Weissnat", username: "Elwyn.Skiles", email: "Telly.Hoeger@billy.biz",…}
7: {id: 8, name: "Nicholas Runolfsdottir V", username: "Maxime_Nienow", email: "Sherwood@rosamond.me",…}
8: {id: 9, name: "Glenna Reichert", username: "Delphine", email: "Chaim_McDermott@dana.io",…}
9: {id: 10, name: "Clementina DuBuque", username: "Moriah.Stanton", email: "Rey.Padberg@karina.biz",…}
2 / 10 requests | 4.7 KB / 11.5 KB transferred | 4.1 KB / 4.2 MB resources |...

การส่งข้อมูลไปยัง Server ผ่าน POST method

Software Park Thailand
</Code Camp>

ใช้ POST ในสถานการณ์อะไรได้บ้าง

- ต้องการส่งข้อมูลกลับไปบันทึกที่ฝั่ง Backend
- ต้องการส่งข้อมูลกลับไปแก้ไขฝั่ง Backend
- ข้อมูลมีขนาดใหญ่และซับซ้อน เช่น JSON object มีหลาย property และขนาดใหญ่ เช่นข้อมูลหน้า Form เพราะฉะนั้นต้องใช้ POST ส่งกลับไป Backend

```
const data = {
};
```

JSON object

การส่งข้อมูลไปยัง server ผ่าน Ajax จะต้องส่ง parameter เพิ่มเติม หลัง url

```
this.httpClient.post('<url>',data).subscribe(result=>{
    //
});
```

Software Park Thailand
</Code Camp>

Lab 4 – ส่งข้อมูลไปที่ API ผ่าน POST

ทำต่อจาก lab 3 **แก้ไข PostListComponent** เพื่อให้รองรับการเพิ่มข้อมูล โดยการสร้าง ReactiveForm ที่หน้า template ดังนี้

ตัว JSON Placeholder รองรับ method POST ด้วย

Routes

All HTTP methods are supported. You can use http or https for your requests.

GET	/posts
GET	/posts/1
GET	/posts/1/comments
GET	/comments?postId=1
POST	/posts
PUT	/posts/1
PATCH	/posts/1
DELETE	/posts/1

1

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';
import { HttpClientModule } from '@angular/common/http';
import { PostListComponent } from './post-list/post-list.component';

import { FormsModule, ReactiveFormsModule } from '@angular/forms';

@NgModule({
 declarations: [AppComponent, PostListComponent],
 imports: [BrowserModule, HttpClientModule, FormsModule, ReactiveFormsModule],
 providers: [],
 bootstrap: [AppComponent]
})
export class AppModule {}
```

2

post-list.component.ts

```
import { Component, OnInit } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { FormBuilder, FormGroup } from '@angular/forms';

@Component({
 selector: 'app-post-list',
 templateUrl: './post-list.component.html',
 styleUrls: ['./post-list.component.css']
})
export class postListComponent implements OnInit {
 posts: any[];
 form: FormGroup;
 constructor(private httpClient: HttpClient, private fb: FormBuilder) {
   this.form = this.fb.group({
     name: '',
     email: ''
   });
 }

 ngOnInit() {
   this.httpClient
     .get('http://codecamp3-simple-api.herokuapp.com/api/posts')
     .subscribe(result => {
       this.posts = result as any[];
     });
 }
}
```

3

..... (Code เหมือน slide ที่แล้วต่อย่อไว้)

post-list.component.ts

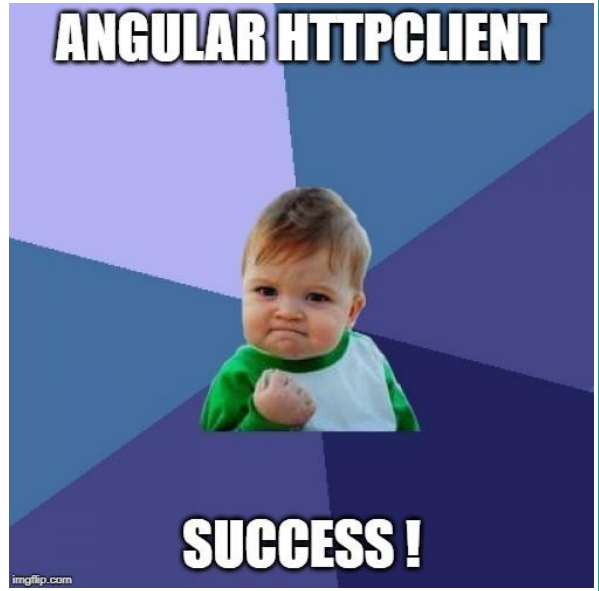
```
export class postListComponent implements OnInit {
 posts: any[];
 form: FormGroup;
 constructor(private httpClient: HttpClient, private fb: FormBuilder) {
   this.form = this.fb.group({
     userId: '',
     title: '',
     body: ''
   });
 }
 ngOnInit() {
   this.loadPost();
 }
 loadPost() {
   this.posts = [];
   this.httpClient
     .get('https://jsonplaceholder.typicode.com/posts')
     .subscribe(result => {
       this.posts = result as any[];
     });
 }
 addPost() {
   const newPost = this.form.value;
   this.httpClient
     .post('https://jsonplaceholder.typicode.com/posts', newPost)
     .subscribe(result => {
       this.form.reset();
       alert('Add Post Success !');
       this.loadPost();
     });
 }
}
```

4

post-list.component.html

```
<form [formGroup]="form">
 <h3>Add New Post</h3>
<div>
   UserId: <input formControlName="userId">
 </div>
 <div>
   Title: <input formControlName="title">
 </div>
 <div>
   Body: <input formControlName="body">
 </div>
 <div>
   <button (click)="addPost()">Add</button>
 </div>
</form>
<div *ngFor="let post of posts" style="border:1px solid grey;margin:4px;padding:4px;">
 <h4>User Id: #{{post.userId}}</h4>
 <span>Id: {{post.id}}</span>
 <span>Title: {{post.title}}</span>
 <span>Body: {{post.body}}</span>
 </div>
```

5



Best Practices

ใช้ dash (-) ในการแยกคำเมื่อใช้คำสั่ง Angular CLI

```
ng g component mylearningcomponent
```

```
ng g component my-learning
```

ตั้งชื่อ method ให้สื่อความหมาย ยาวได้ไม่เป็นไร

```
<button (click)=”fooBar()”>
```

```
<button (click)=”submitRegistration()”>
```

ตั้งชื่อ property ให้ใช้คำนาม (noun)

```
getData : string;
```

```
data : string
```

ตั้งชื่อ method ให้ใช้คำกิริยา (verb)

```
data(){

}
```

```
getData(){

}
```

ตั้งชื่อ property ให้สื่อความหมาย ยาวได้ไม่เป็นไร

```
a : string;
```

```
accessToken : string
```

ตั้งชื่อ file pattern ให้ตรงกับชนิดของ source code ในไฟล์

```
foo.ts
bar.ts
abc.ts
```

```
foo.component.ts
bar.service.ts
abc.module.ts
```

ใช้ dash (-) ในการแยกคำและต้องเป็นตัวพิมพ์เล็กในการตั้งชื่อไฟล์

```
MyComponent.component.ts
memberList.Component.ts
ProductListService.ts
```

```
my-component.component.ts
member-list.component.ts
product-list.module.ts
```

ตั้งชื่อตัวแปรแบบพหุพจน์สำหรับ array

```
item : string[];
task : Task[];
```

```
items : string[];
tasks : Task[];
```

Angular Style Guide

https://angular.io/guide/styleguide

Software Park Thailand
</Code Camp>

Angular from Zero to Hero

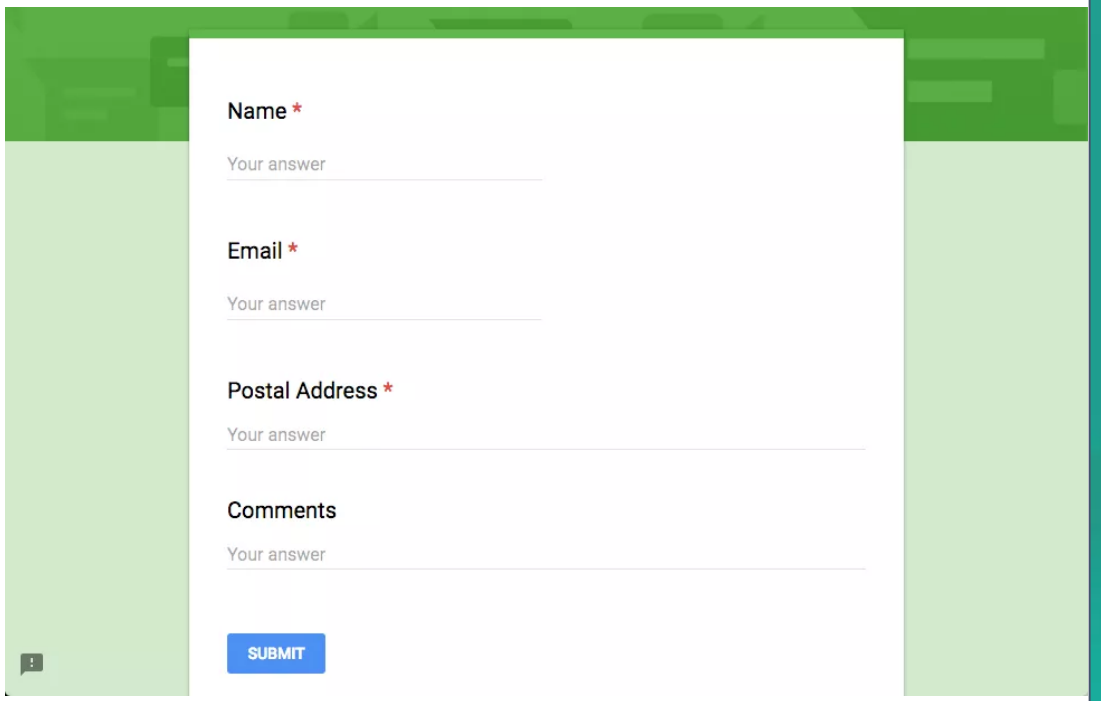
- Modularity application
- Inter-Component communication and Events
- Lazy Loading Routing
- Custom Validation
- Dynamic Reactive Form
- RxJS

Homework

Software Park Thailand
</Code Camp>

Homework 1

- **ใช้ Bootstrap CSS ทำ form**
- สร้าง form ตามภาพ
- required fieds ให้ใส่ * สีแดงด้านหลัง
- email ต้อง validate ว่าถูก pattern หรือไม่



Software Park Thailand
</Code Camp>

Homework 2

- ทำโปรแกรมคำนวนค่าเงิน (Exchange Rate)
- ใช้ API จาก https://exchangeratesapi.io/
- ระบุ code, จำนวนเงิน ประเทศต้นทาง และ code ปลายทางได้ (เป็น textbox) เพื่อให้ได้จำนวนเงินปลายทาง