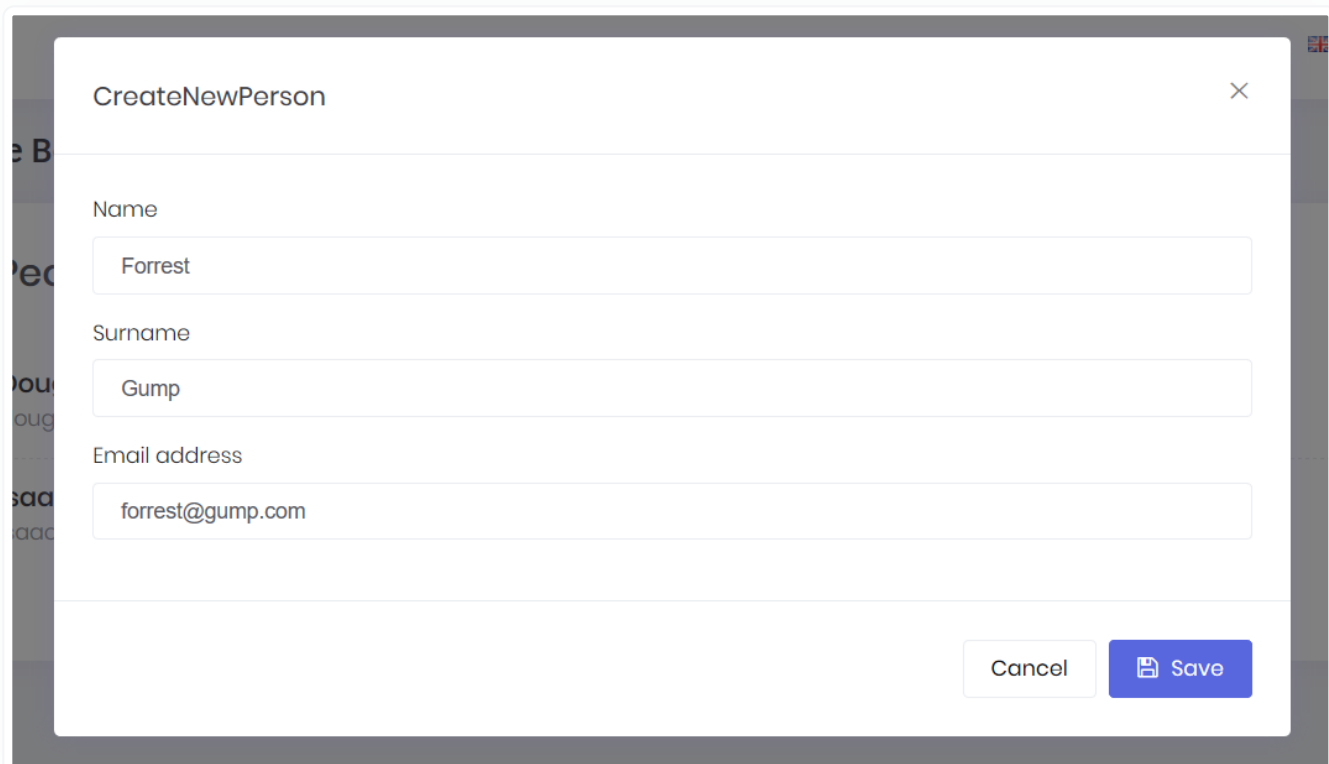




Creating Modal for New Person

We will create a Bootstrap Modal to create a new person. ASP.NET Zero uses [ngx-bootstrap](#) library to create modals (you can use another library, but we will use it in this sample too). Final modal will be like below:



First of all, we should use **nswag/refresh.bat** to re-generate service-proxies. This will generate the code that is needed to call **PersonAppService.CreatePerson** method from client side. Notice that you should rebuild & run the server side application before re-generating the proxy scripts.

We are starting from creating a new component, named **create-person-modal.component.ts** into client side phonebook folder:

```
import { Component, ViewChild, Injector, ElementRef, Output, EventEmitter } from '@angular/core';
import { ModalDirective } from 'ngx-bootstrap';
import { PersonServiceProxy, CreatePersonInput } from '@shared/service-proxies/service-proxies';
import { AppComponentBase } from '@shared/common/app-component-base';
import { finalize } from 'rxjs/operators';
```



ASP.NET CORE ANGULAR

DOCUMENTS

```
templateUrl: './create-person-modal.component.html'
})
export class CreatePersonModalComponent extends AppComponentBase {

    @Output() modalSave: EventEmitter<any> = new EventEmitter<any>();

    @ViewChild('modal' , { static: false }) modal: ModalDirective;
    @ViewChild('nameInput' , { static: false }) nameInput: ElementRef;

    person: CreatePersonInput = new CreatePersonInput();

    active: boolean = false;
    saving: boolean = false;

    constructor(
        injector: Injector,
        private _personService: PersonServiceProxy
    ) {
        super(injector);
    }

    show(): void {
        this.active = true;
        this.person = new CreatePersonInput();
        this.modal.show();
    }

    onShown(): void {
        this.nameInput.nativeElement.focus();
    }

    save(): void {
        this.saving = true;
        this._personService.createPerson(this.person)
            .pipe(finalize(() => this.saving = false))
            .subscribe(() => {
                this.notify.info(this.l('SavedSuccessfully'));
                this.close();
                this.modalSave.emit(this.person);
            });
    }

    close(): void {
```



}

Let me explain some parts of this class:

- It has a selector, **createPersonModal**, which will be used as like an HTML element in the person list page.
- It extends **AppComponentBase** to take advantage of it (defines this.l and this.notify in this sample).
- Defines an event, **modalSave**, which is triggered when we successfully save the modal. Thus, the main page will be informed and it can reload the person list.
- Declares two **ViewChild** members (**modal** and **nameInput**) to access some elements in the view.
- Injects **PersonServiceProxy** to call server side method while creating the person.
- It focuses to **name** input when modal is shown.

The code is simple and easy to understand except a small hack: an active flag is used to reset validation for Angular view (explained in angular's [documentation](#)).

As declared in the component, we are creating the **create-person-modal.component.html** file in the same folder as shown below:

```
<div bsModal #modal="bs-modal" (onShown)="onShown()" class="modal fade" tabindex="-1">
  <div class="modal-dialog modal-lg">
    <div class="modal-content">
      <form *ngIf="active" #personForm="ngForm" novalidate (ngSubmit)="save()"
        <div class="modal-header">
          <h4 class="modal-title">
            <span>{{"CreateNewPerson" | localize}}</span>
          </h4>
          <button type="button" class="close" (click)="close()" aria-label="Close">
            <span aria-hidden="true">&times;</span>
          </button>
        </div>
        <div class="modal-body">
          <div class="form-group">
            <label>{{"Name" | localize}}</label>
            <input #nameInput class="form-control" type="text" name="name">
          </div>
          <div class="form-group">
            <label>{{"Surname" | localize}}</label>
            <input class="form-control" type="email" name="surname" [(ngModel)]="surname">
          </div>
          <div class="form-group">
            <label>{{"Email" | localize}}</label>
            <input class="form-control" type="email" name="email" [(ngModel)]="email">
          </div>
          <div class="form-group">
            <label>{{"Phone" | localize}}</label>
            <input class="form-control" type="text" name="phone" [(ngModel)]="phone">
          </div>
          <div class="form-group">
            <label>{{"Address" | localize}}</label>
            <input class="form-control" type="text" name="address" [(ngModel)]="address">
          </div>
          <div class="form-group">
            <label>{{"City" | localize}}</label>
            <input class="form-control" type="text" name="city" [(ngModel)]="city">
          </div>
          <div class="form-group">
            <label>{{"State" | localize}}</label>
            <input class="form-control" type="text" name="state" [(ngModel)]="state">
          </div>
          <div class="form-group">
            <label>{{"Zip" | localize}}</label>
            <input class="form-control" type="text" name="zip" [(ngModel)]="zip">
          </div>
          <div class="form-group">
            <label>{{"Save" | localize}}</label>
            <button type="button" class="btn btn-primary" (click)="save()">
              <span>{{"Save" | localize}}</span>
            </button>
          </div>
        </div>
      </form>
    </div>
  </div>
</div>
```

ASP.NET CORE ANGULAR

DOCUMENTS

```
</div>
<div class="modal-footer">
  <button [disabled]="saving" type="button" class="btn btn-second
  <button type="submit" class="btn btn-primary" [disabled]="!pers
</div>
</form>
</div>
</div>
</div>
```

Most of this code is similar for all modals. The important part is how we binded model to the view using the ngModel directive. As like all components, Angular requires to relate it to a module. We should add it to **declarations** array of **phonebook.module.ts** as like shown below:

```
import {NgModule} from '@angular/core';
import {AppSharedModule} from '@app/shared/app-shared.module';
import {PhoneBookRoutingModule} from './phonebook-routing.module';
import {PhoneBookComponent} from './phonebook.component';
import {CreatePersonModalComponent} from './create-person-modal.component';

@NgModule({
  declarations: [PhoneBookComponent, CreatePersonModalComponent],
  imports: [AppSharedModule, PhoneBookRoutingModule]
})
export class PhoneBookModule {}
```

We need to put a "Create new person" button to the 'people list page' to open the modal when clicked to the button. To do that, we made the following changes in **phonebook.component.html**:

```
<div [@routerTransition]>
  <div class="kt-content kt-grid__item kt-grid__item--fluid kt-grid kt-grid--hor
    <div class="kt-subheader kt-grid__item">
      <div class="kt-container ">
        <div class="kt-subheader__main">
          <h3 class="kt-subheader__title">
            <span>{"PhoneBook" | localize}</span>
          </h3>
          <span class="kt-subheader__separator kt-subheader__separator--v
          <span class="kt-subheader__desc">
```





ASP.NET CORE ANGULAR

DOCUMENTS

```
<div class="kt-subheader__toolbar">
  <div class="kt-subheader__wrapper">
    <button class="btn btn-primary" (click)="createPersonModal.
      {"CreateNewPerson" | localize}}></button>
  </div>
</div>
</div>
</div>
<div class="kt-container kt-grid__item kt-grid__item--fluid">
  <div class="kt-portlet kt-portlet--mobile">
    <div class="kt-portlet__body kt-portlet__body--fit">
      <h3>{{"AllPeople" | localize}}</h3>
      <div class="row kt-row--no-padding align-items-center" *ngFor="
        <div class="col">
          <h4>{{person.name + ' ' + person.surname}}</h4>
          <span>{{person.emailAddress}}</span>
        </div>
      </div>
    </div>
  </div>
</div>
<createPersonModal #createPersonModal (modalSave)="getPeople()"></creat.
</div>
</div>
</div>
```

Feedback



Made some minor changes in the view; Added a **button** to open the modal and the **createPersonModal** component as like another HTML tag (which matches to the selector in the **create-person-modal.component.ts**).

Next

- [Authorization For Phone Book](#)