

Mục lục

1. Khái niệm.....	2
2. Cài đặt môi trường phát triển.....	2
3. Cấu trúc thư mục.....	2
4. Cơ chế navigation: NavController	3
- Ví dụ 1: chuyển trang đơn giản.....	3
- Ví dụ 2: chuyển trang có truyền tham số	3
- Ví dụ 3: Chuyển sang trang chi tiết.....	4
- Ví dụ 4: Trở về trang trước.....	5
- Ví dụ 5: Trở về trang chủ.....	5
- Chuyển trang sử dụng directive	5
- Chuyển trang sử dụng animation : (Tham số thứ 3 của NavController.push)	5
5. Các component thường sử dụng	5
a. Button.....	5
b. List.....	6
c. ActionSheet	7
d. Alert.....	9
e. Badges	14
f. Toast	14
g. Loading	14
h. Grid.....	15
i. List mở rộng	15
j. Slide	15
k. Items.....	16
l. Segment.....	16
m. card	17

IONIC

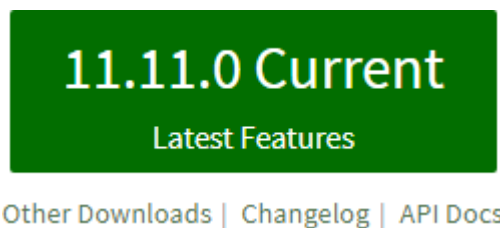
1. Khái niệm

- Ionic framework là một framework sử dụng HTML5 và Angular để xây dựng các ứng dụng mobile chạy đa nền tảng.
- Ứng dụng xây dựng bằng ionic framework có thể chạy được trên web, trên android, trên ios
- Xây dựng một lần duy nhất và sau này có thể chạy trên tất cả nền tảng
- Trang chủ của ionic framework: <https://ionicframework.com/>

2. Cài đặt môi trường phát triển

a. Cài đặt nodeJS:

- vào trang chủ của nodeJS, download và tiến hành cài đặt <https://nodejs.org/en/>



- Kiểm tra cài đặt thành công hay chưa: **node -v**

b. Cài đặt ionic-cordova CLI

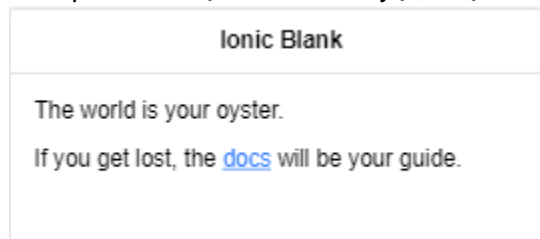
- Đây là công cụ hỗ trợ thực hiện tạo project, tạo page, tạo component ở trên một project ionic
- Chạy lệnh **npm install -g ionic cordova** để tiến hành cài đặt ionic và cordova

c. Tạo project mới

- **ionic start tên_project loại_giao_diện**
- có các loại giao diện là: **blank, tabs, sidemenu...**

d. Khởi chạy project:

- **ionic serve (--port ????)**
- Kết quả hiển thị trên trình duyệt, chọn môi trường ios:



3. Cấu trúc thư mục

- Trong project vừa tạo, cũng có các thành phần tương tự như của Angular
- File [src/index.html](#) là file chính của dự án
- Mục đích của nó là thiết lập các tập lệnh, khai báo CSS, bootstrap và component root
- Toàn bộ code của dự án được viết trong thư mục **./src/**
- File **app.module.ts** là module chạy đầu tiên của ứng dụng
 1. Trong file này có phần khai báo các Page, component, service...

2. Khi khai báo một page thì cũng khai báo vào **entryComponent**
 3. Mảng providers dùng để khai báo các service
 4. Bootstrap: khai báo component chạy đầu tiên
- Thư mục assets để khai báo các thành phần CSS, các hình ảnh...
 - Thư mục theme chứa file **variables.scss**, dùng để định nghĩa các tài nguyên

4. Cơ chế navigation: NavController

- Ionic không sử dụng Router như trong Angular
 - Sử dụng navigation để chuyển trang theo kiểu ngăn xếp (Stack of Pages)
 - Ví dụ 1: chuyển trang đơn giản
1. Tạo trang mới: **ionic g page users**, một thư mục mới được tạo ra chứa 4 file: html, scss, module.ts và ts. Trong 1 trang gồm có các thành phần: **header**, **content**, **footer**
 2. Từ trang **home**, muốn chuyển sang trang **users**, ta làm như sau:
 - a) Trong file **home.html**, tạo một button để khi kích vào thì thực hiện hàm **goToUsers**

```
<button ion-button color="danger" (click)="goToUsers()">Danh sách người dùng</button>
```

b) Trong file **home.ts**,

- Import page **users** vào

```
import { UsersPage } from '../users/users'
```

- định nghĩa hàm **goToUsers**, để chuyển sang trang users

```
goToUsers(){
  this.navCtrl.push(UsersPage);
}
```

c) Kết quả, ở trang home, khi nhấn vào button thì sẽ chuyển sang trang **users**

- Ví dụ 2: chuyển trang có truyền tham số

1. Trong file **home.ts**, tạo danh sách **users**:

```
users: any[] = [
  {
    name: "Nguyen Van Dinh",
    age: 19
  },
  {
    name: "Le Thi Hoa",
    age: 20
  },
  {
    name: "Tran Thi Hao",
    age: 18
  }
]
```

2. Để chuyển sang trang users, có truyền tham số là danh sách users, thêm thuộc tính này vào hàm **goToUsers**

```
goToUsers(){
  this.navCtrl.push(UsersPage, this.users);
}
```

3. ở file **users.ts**, nhận giá trị này bằng hàm **ngOnInit**, như sau:

```
ngOnInit(){
  this.users = this.navParams.data;
}
```

4. Ở file **users.html**, dùng ***ngFor** để lặp danh sách **users** và hiển thị ra trình duyệt

```
<ng-container *ngFor="let user of users">
  <h4>{{ user.name }}</h4>
  <p>{{ user.age }}</p>
```

5. kết quả, khi nhấn button chuyển sang trang **users** thì hiển thị danh sách users

6. Nếu truyền tham số là một object:

```
goToUsers(){
  this.navCtrl.push(UsersPage, { list: this.users });
}
```

Hoặc

```
goToUsers(){
  let param = { list: this.users };
  this.navCtrl.push(UsersPage, param);
}
```

7. Nhận tham số như sau:

```
ngOnInit(){
  this.users = this.navParams.get("list");
}
```

- **Ví dụ 3:** Tạo trang user detail, để khi kích vào từng user ở list users thì sẽ chuyển sang thông tin chi tiết của một user

1. Tạo trang mới user: **ionic g page user**

2. Tại file users.html, tạo một sự kiện, để khi kích vào thì chuyển sang trang chi tiết

```
<h4 (click)="goToUserDetail(user)" ion-button>{{ user.name }}</h4>
```

3. Tại file users.ts, định nghĩa hàm **goToUserDetail** để chuyển sang trang user

```
goToUserDetail(user) {
  this.navCtrl.push(UserPage, user);
}
```

4. Tại file user.ts, dùng hàm **ngOnInit** để nhận dữ liệu từ Users chuyển sang

```
ngOnInit() {
  this.user = this.navParams.data;
}
```

5. Hiển thị thông tin ra ở file user.html

```
<h3>{{ user.name }}</h3>
<p>{{ user.age }}</p>
```

- **Ví dụ 4: Trở về trang trước**

- Mặc định **NavController** tạo một button back để trở lại trang trước
- Mở rộng: trở về trang trước bằng cách khác

1. **NavController.pop()**

2. **NavController.popToRoot()**

- **Ví dụ áp dụng:**

1. Tại file **user.html**, tạo sự kiện để khi kích vào thì trở về trang users

```
<button ion-button (click)="goToUsers()"> Trở về trang Users</button>
```

2. Tại file **user.ts**, định nghĩa hàm **goToUsers**

```
goToUsers() {  
  this.navCtrl.pop()  
}
```

- **Ví dụ 5: Trở về trang chủ**

1. Tại file **user.html**, tạo sự kiện để khi kích vào thì trở về trang chủ

```
<button ion-button (click)="goToHome()"> Trở về trang chủ</button>
```

2. Tại file **user.ts**, định nghĩa hàm **goToHome**

```
goToHome() {  
  this.navCtrl.popToRoot();  
}
```

- **Chuyển trang sử dụng directive**

1. Tạo đối tượng thuộc trang cần chuyển => [navPush] = “đối tượng trang cần chuyển”
2. Truyền tham số: [navParams]=”data”

- Trở về: sử dụng directive: navPop

- **Chuyển trang sử dụng animation: (Tham số thứ 3 của NavController.push)**

1. direction: “back” hoặc “forward” (mặc định) => nếu dùng direction: “back” thì sẽ gỡ trang hiện tại ra và gắn trang mới vào
2. duration: 2000 => thời gian chuyển trang
3. easing: “ease-out”/”ease-in”

```
this.navCtrl.push(UsersPage, param, {  
  direction: "back",  
  duration: 2000,  
  easing: "ease-in"  
});
```

5. Các component thường sử dụng

Truy cập trang chủ của ionic framework để sử dụng các component:

<https://ionicframework.com/docs/v3/components/>

a. Button

- **Button** được tạo ra bằng thẻ **button** trong html kết hợp với directive **ion-button** của ionic 3
- Thông qua button có thể bắt các sự kiện và xử lý bằng event binding, ví dụ click, double click, long click,...

- Để thay đổi màu sắc button ta sử dụng thuộc tính **color**, các biến chứa mã màu được lưu trữ tại **theme/variables.scss**. Các màu cơ bản: **primary, secondary, danger, light, dark**. Mặc định **primary**
- Thuộc tính **outline**: tạo ra button chỉ có đường viền
- Thuộc tính **clear**: xóa toàn bộ màu viền và màu nền
- Thuộc tính **round**: tạo button có đường viền bo tròn ở bốn góc
- Thuộc tính **block**: hiển thị theo 1 hàng độc lập
- Thuộc tính **full**: hiển thị theo 1 hàng độc lập tràn ra 2 biên
- Thuộc tính chỉnh kích cỡ: **small, medium, large**
- Thêm **<ion-icon>** vào button, các thuộc tính:
 - **icon-start**: icon ở đầu button
 - **icon-end**: icon ở cuối button
 - **icon-only**: button chỉ có icon, icon to hơn

b. ActionSheet

- **ActionSheet** là bảng thông báo trượt lên từ cạnh dưới của thiết bị và hiển thị các tùy chọn
- Các thành phần: **title, cssClass, buttons** (trong **buttons** có các thành phần như **text, handler, icon...**)
- Muốn sử dụng **ActionSheet**, import vào

```
import { ActionSheetController } from 'ionic-angular';
```

- Khởi tạo: `constructor(public actionsheetCtrl: ActionSheetController) {}`

- Ví dụ:

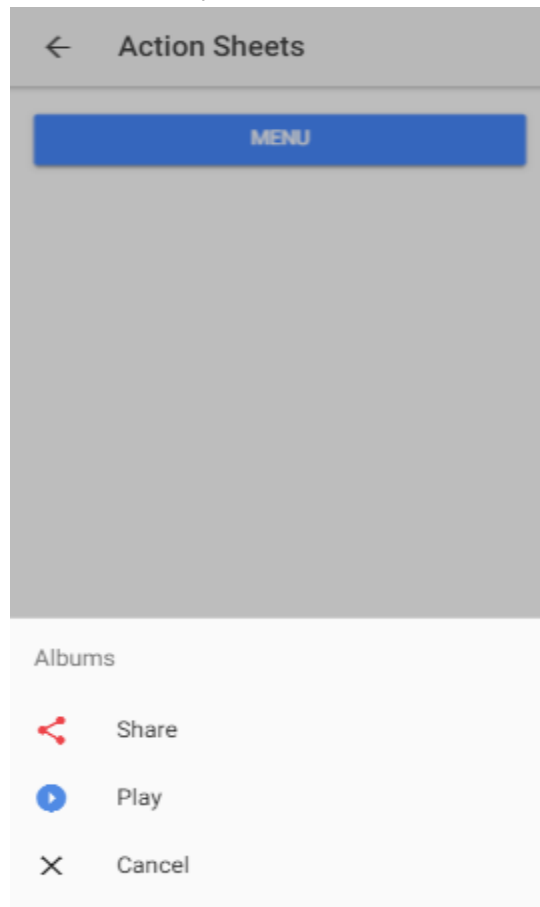
```
openMenu() {
  let actionSheet = this.actionsheetCtrl.create({
    title: 'Albums',
    cssClass: 'action-sheets-basic-page',
    buttons: [
      {
        text: 'Delete',
        icon: !this.platform.is('ios') ? 'trash' : null,
        handler: () => {
          console.log('Delete clicked');
        }
      },
      {
        text: 'Share',
        icon: !this.platform.is('ios') ? 'share' : null,
        handler: () => {
          console.log('Share clicked');
        }
      },
      {
        text: 'Play',
        icon: !this.platform.is('ios') ? 'arrow-dropright-circle' : null,
        handler: () => {
          console.log('Play clicked');
        }
      },
      {
        text: 'Favorite',
```

```

    icon: !this.platform.is('ios') ? 'heart-outline' : null,
    handler: () => {
      console.log('Favorite clicked');
    }
  },
  {
    text: 'Cancel',
    role: 'cancel', // will always sort to be on the bottom
    icon: !this.platform.is('ios') ? 'close' : null,
    handler: () => {
      console.log('Cancel clicked');
    }
  },
]
});
actionSheet.present();
}

```

- Kết quả hiển thị ra trình duyệt:

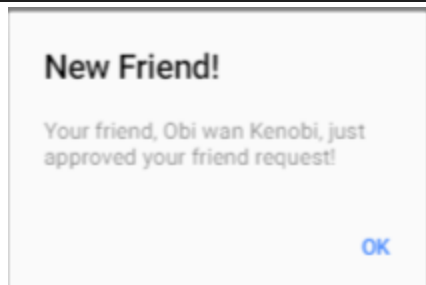


c. Alert

- Alert cung cấp bảng chọn để người dùng xác nhận hoặc có nhiều tùy chọn khác nhau
- Các thành phần: **Prompt Alerts**, **Confirmation Alerts**, **Radio Alerts**, **Checkbox Alerts**

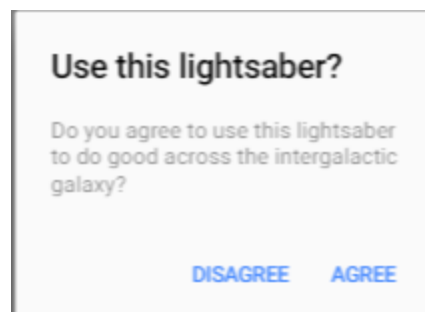
1. Alert chỉ có nút chọn OK

```
doAlert() {  
  let alert = this.alerCtrl.create({  
    title: 'New Friend!',  
    message: 'Your friend, Obi wan Kenobi, just approved your friend request!',  
    buttons: ['Ok']  
  });  
  alert.present()  
}
```



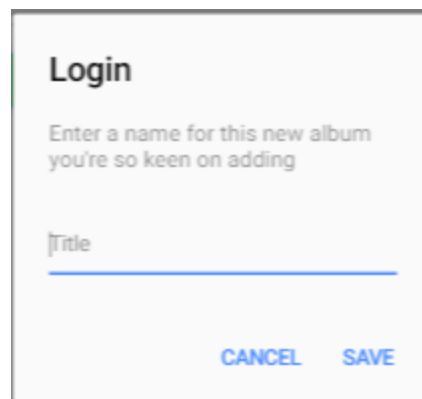
2. Alert để xác nhận (Đồng ý hoặc không?)

```
doConfirm() {  
  let confirm = this.alerCtrl.create({  
    title: 'Use this lightsaber?',  
    message: 'Do you agree to use this lightsaber to do good across the i  
ntergalactic galaxy?',  
    buttons: [  
      {  
        text: 'Disagree',  
        handler: () => {  
          console.log('Disagree clicked');  
        }  
      },  
      {  
        text: 'Agree',  
        handler: () => {  
          console.log('Agree clicked');  
        }  
      }  
    ]  
  });  
  confirm.present()  
}
```



3. Alert để nhập thông tin

```
doPrompt() {  
  let prompt = this.alertCtrl.create({  
    title: 'Login',  
    message: "Enter a name for this new album you're so keen on adding",  
    inputs: [  
      {  
        name: 'title',  
        placeholder: 'Title'  
      }  
    ],  
    buttons: [  
      {  
        text: 'Cancel',  
        handler: data => {  
          console.log('Cancel clicked', data);  
        }  
      },  
      {  
        text: 'Save',  
        handler: data => {  
          console.log('Saved clicked', data);  
        }  
      }  
    ]  
  });  
  prompt.present();  
}
```



Login

Enter a name for this new album
you're so keen on adding

CANCEL SAVE

4. Alert hiện các radio để chọn (chỉ được chọn 1)

```
doRadio() {  
  let alert = this.alerCtrl.create();  
  alert.setTitle('Lightsaber color');
```

```
  alert.addInput({  
    type: 'radio',  
    label: 'Blue',  
    value: 'blue',  
    checked: true  
  });
```

```
  alert.addInput({  
    type: 'radio',  
    label: 'Green',  
    value: 'green'  
  });
```

```
  alert.addInput({  
    type: 'radio',  
    label: 'Red',  
    value: 'red'  
  });
```

```
  alert.addInput({  
    type: 'radio',  
    label: 'Yellow',  
    value: 'yellow'  
  });
```

```
  alert.addInput({  
    type: 'radio',  
    label: 'Purple',  
    value: 'purple'  
  });
```

```
  alert.addInput({  
    type: 'radio',  
    label: 'White',  
    value: 'white'  
  });
```

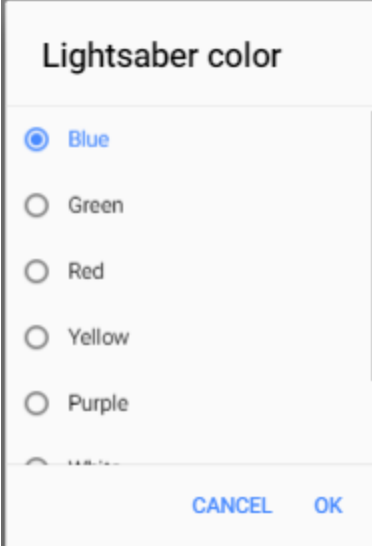
```
  alert.addInput({  
    type: 'radio',  
    label: 'Black',  
    value: 'black'  
  });
```

```
  alert.addButton('Cancel');
```

```
  alert.addButton({  
    text: 'Ok',  
    handler: data => {
```

```
        console.log('Radio data:', data);
        this.RadioResult = data;
    }
});

alert.present().then(() => {
    console.log('Radio Open', this.RadioResult)
});
}
```



Lightsaber color

☒ Blue

☐ Green

☐ Red

☐ Yellow

☐ Purple

☐ White

CANCEL OK

5. Alert hiện các checkbox để chọn (0, 1 hoặc nhiều)

```
doCheckbox() {  
  let alert = this.alertCtrl.create();  
  alert.setTitle('Which planets have you visited?');  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Alderaan',  
    value: 'value1',  
    checked: true  
  });  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Bespin',  
    value: 'value2'  
  });  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Coruscant',  
    value: 'value3'  
  });  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Endor',  
    value: 'value4'  
  });  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Hoth',  
    value: 'value5'  
  });  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Jakku',  
    value: 'value6'  
  });  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Naboo',  
    value: 'value6'  
  });  
  
  alert.addInput({  
    type: 'checkbox',  
    label: 'Takodana',  
    value: 'value6'  
  });  
}
```

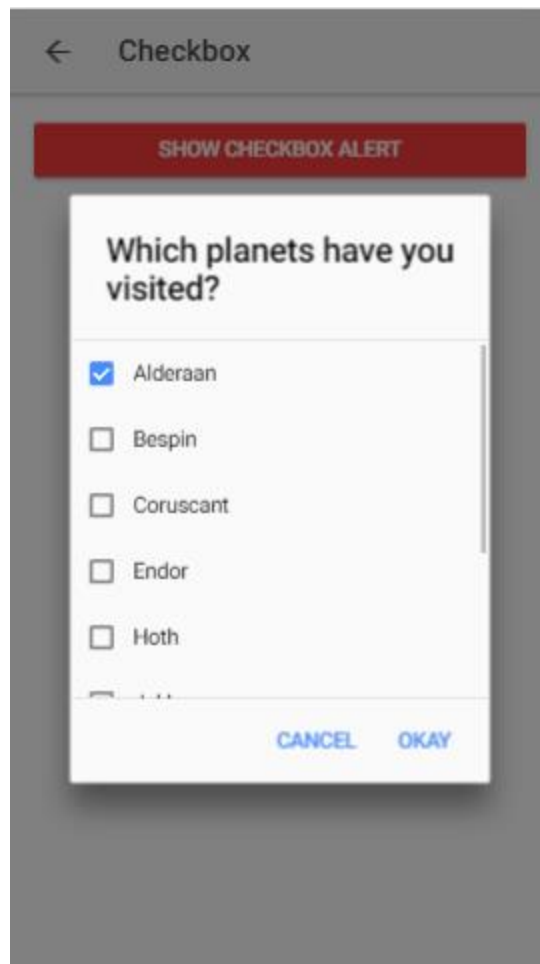
```

    });

    alert.addInput({
      type: 'checkbox',
      label: 'Tatooine',
      value: 'value6'
    });

    alert.addButton('Cancel');
    alert.addButton({
      text: 'Okay',
      handler: data => {
        console.log('Checkbox data:', data);
        this.CheckboxResult = data;
      }
    });
    alert.present().then(() => {
      console.log('Open Checkbox: ', this.CheckboxResult);
    });
  }
}

```



d. List

- Dùng để hiển thị các dòng thông tin: danh bạ, danh sách nhạc, menu
- Cú pháp: `<ion-list>`
- Theo mặc định, list sẽ có đường ngăn giữa các thành phần,
- Mỗi thành phần `<button ion-item>Nguyễn Văn Định</button>` là một nút có mũi tên bên phải ở chế độ ios

1. Avatars

2. Dividers

3. Headers

4. Icons

5. Inset: directive inset để thêm lề trái và phải cho list

6. Multiline

7. No-lines

8. Sliding

9. Thumbnail

e. Badges

- **Badges** là các thành phần nhỏ thường truyền đạt một giá trị số cho người dùng.
- Chúng thường được sử dụng trong `<ion-item>`

Ví dụ:

```
<ion-badge item-end>260k</ion-badge>
```

260k

f. Toast

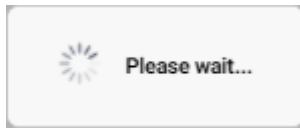
- Toast là một thông báo xuất hiện trên đầu nội dung của ứng dụng.
- Thông báo Toast sẽ tự động mất đi sau một khoảng thời gian nhất định
- Các thành phần:
 - o **Message**: thông báo
 - o **Duration**: thời gian diễn ra (ms)
 - o **Position**: vị trí (top/bottom/middle)
 - o **showCloseButton**: hiện nút để click close (true/false)
 - o **closeButtonText**: text cho nút close

```
showToast(position: string) {  
  let toast = this.toastCtrl.create({  
    message: 'Mmmm, buttered toast',  
    duration: 2000,  
    position: position  
  });  
  
  toast.present(toast);  
}
```

g. Loading

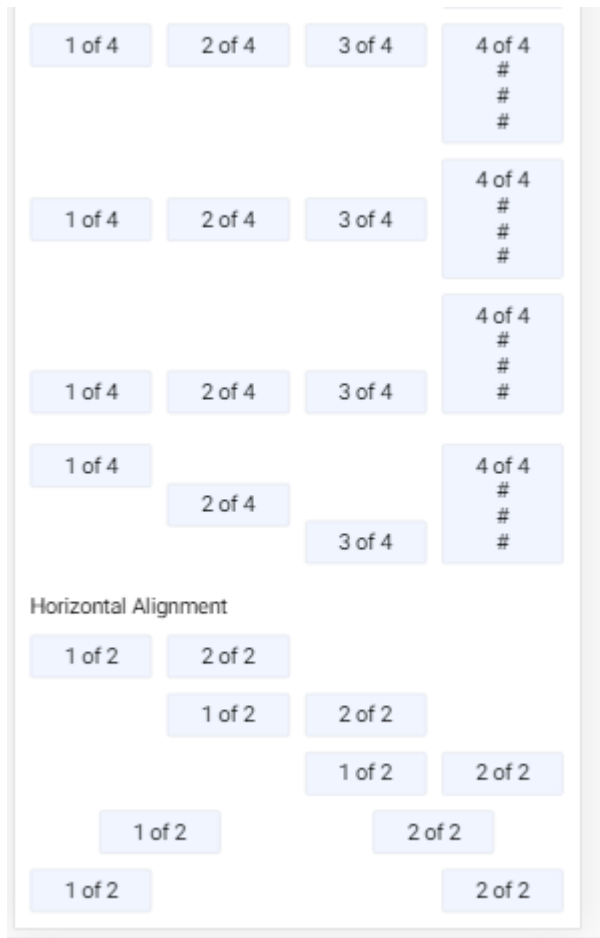
- **Loading** là lớp phủ ngăn chặn sự tương tác của người dùng trong khi chờ ra hoạt động
- **Loading** diễn ra khi đang trong quá trình tải dữ liệu,...
- Các thành phần:
 - o Content: nội dung hiển thị
 - o Duration: thời gian hiển thị
 - o spinner: 'hide' => ẩn biểu tượng loading
- Dùng lệnh **loading.dismiss()**; để dừng loading
- Hàm **loading.onDidDismiss()** thực hiện lệnh sau khi đã loading xong
- Lệnh **loading.present()** để thực thi lệnh loading

```
presentLoading() {  
  this.loadingCtrl.create({  
    content: 'Please wait...',  
    duration: 3000  
  }).present();  
}
```



h. Grid

- Hệ thống lưới được tạo thành từ 12 cột
- Mặc định thì hệ thống lưới sẽ chia đều bề rộng cho số cột và chiếm hết 12 ô
- Dùng **col-<width>** để đặt bề rộng cho một cột
- Dùng **col-auto** để cột được đặt bề rộng tùy theo bề rộng của nội dung
- Dùng **offset-<width>** để đặt một khoảng cách phía trước cột
- Ngoài ra còn có các directive khác như:
 - **align-items-start, align-items-center, align-items-end,**
 - **align-self-start, align-self-center, align-self-end,**
 - **justify-content-start, justify-content-center, justify-content-end, justify-content-around, justify-content-between**



i. List mở rộng

- Sự kiện **reorder="true"** và **ionItemReorder** cho phép dùng chuột kéo/thả để sắp xếp lại danh sách
- Sự kiện **ionRefresh** cho phép refresh lại danh sách khi kéo chuột xuống, sau khi refresh thì danh sách được cập nhật lại
- Sự kiện **ionInfinite** cho phép tải thêm dữ liệu nếu kéo chuột xuống

j. Slide

- Khai báo **@ViewChild(Slides) slides: Slides;** để có biến slides làm việc với slide
- Lệnh **this.slides.slideTo(1, 500);** để chuyển đến slide mong muốn
- Lệnh **this.slides.getActiveIndex();** để lấy chỉ số slide hiện tại
- Lệnh **this.slides.slideNext(500, true)** để chuyển đến slide tiếp theo
- Sự kiện **ionSlideDidChange** thực hiện khi slide được thay đổi
- Lệnh **this.slides.lockSwipes(true);** cho phép chuyển giữa các slide
- Thuộc tính **centeredSlides="true"** để canh giữa slide

- Thuộc tính **autoplay="2000"** để tự động chuyển slide sau 2 giây
- **k. Items**
- Một **item** cơ bản phải được viết dưới dạng phân tử **<ion-item>**
- ion-item có thể được thêm vào <button>
- Theo mặc định, <button> và <a> với thuộc tính ion-item sẽ hiển thị biểu tượng mũi tên phải trên chế độ ios.
- Để ẩn/hiện mũi tên bên phải, thêm các thuộc tính **detail-none/detail-push**
- Thuộc tính **text-wrap** giúp đoạn text tự động xuống hàng khi quá dài
- Thuộc tính **disabled** để vô hiệu hóa các button

I. Segment

- **Segment** là một nhóm các nút, đôi khi được gọi là Điều khiển phân đoạn, cho phép người dùng tương tác với một nhóm nhỏ gọn của một số điều khiển.
- **ngSwitch/ngSwitchCase** để chọn ra các trường hợp điều kiện

```
<ion-header>
  <ion-navbar>
    <ion-title>Badges</ion-title>
  </ion-navbar>
  <ion-toolbar no-border-top>
    <ion-segment [(ngModel)]="pet">
      <ion-segment-button value="q1">
        Quyền 1
      </ion-segment-button>
      <ion-segment-button value="q2">
        Quyền 2
      </ion-segment-button>
    </ion-segment>
  </ion-toolbar>
</ion-header>

<ion-content>
  <div [ngSwitch]="pet">
    <ion-item *ngSwitchCase="'q1'">
      
    </ion-item>
    <ion-item *ngSwitchCase="'q2'">
      
    </ion-item>
  </div>
</ion-content>
```

- Kết quả hiển thị ra trình duyệt



m. card

```
<ion-card>
  <ion-card-title>
    This is title
  </ion-card-title>
  <ion-card-content>
    This is content Individual tabs are just
    This is content Individual tabs are just
    This is content Individual tabs are just
    This is content Individual tabs are just
    This is content Individual tabs are just
  </ion-card-content>
</ion-card>
```

This is title

This is content Individual tabs are just This is
content Individual tabs are just This is content
Individual tabs are just This is content Individual
tabs are just This is content Individual tabs are just