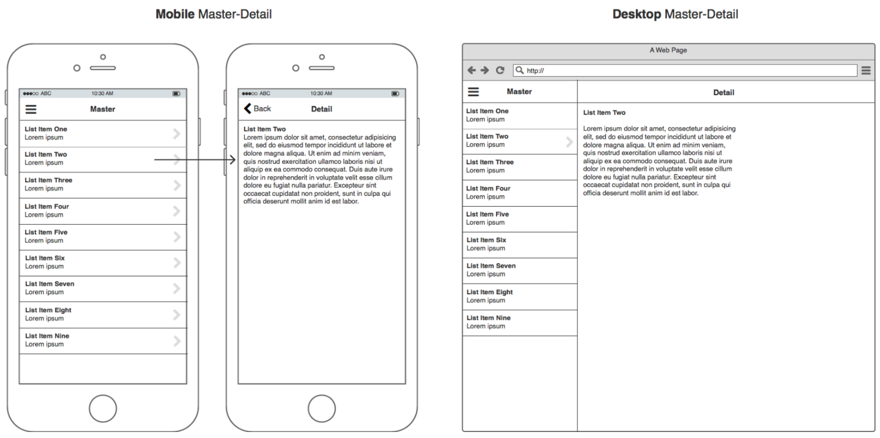
# Ionic 3 > Master-Detail App using the split-pane component

Martin Pritchard
Jun 20, 2017 · 5 min read

Github: DEMO CODE

A little while back, the brilliant guys @ Ionic released a 'split-pane' component that promised to move ionic apps a step closer to being truly responsive, all the way up to the desktop. Whilst their post announcing the new component looked great, all the examples seemed to use the new component to display a menu alongside your main app content. Whilst I'm sure this will be a popular approach, I wanted to use the split-panel to help me implement a 'master-detail' style app, like this…



…and there didn't seem to be any examples out there to help get up and running with this type of thing. So…

## Getting started

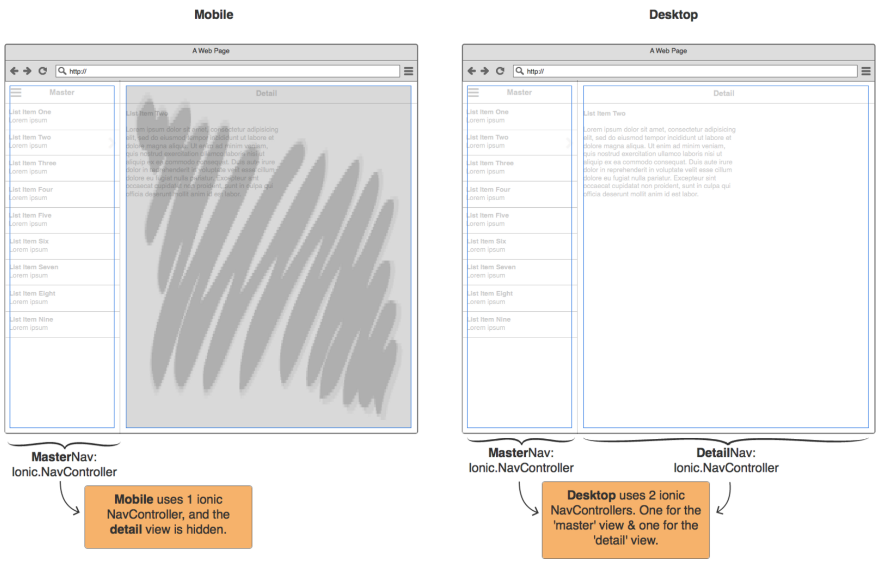
I created a 'blank' ionic 3 project, created using the ionic-cli. To this, I added 3 pages (see github for all demo code):

- An 'Items' page to contain our list items (the 'master view')
- An 'Item' page to display an individual item (the 'detail view')
- A 'Placeholder' page to 'fill the gap' on desktop when no 'item' is selected

Now, whilst getting this to work, the key idea to think about is that:

- On **desktop,** you're going to need **two** ionic NavControllers, one for the master view, and one for the detail view
- On **mobile**, you're only going to need **one** ionic NavController, and your going to want to hide the detail view

So, thats…


Mobile
Desktop
Master
Detail
MasterNav: Ionic.NavController
Mobile uses 1 ionic NavController, and the detail view is hidden.
MasterNav: Ionic.NavController
DetailNav: Ionic.NavController
Desktop uses 2 ionic NavControllers. One for the 'master' view & one for the 'detail' view.

So how do we do this? Well, we'll need to:

- Tweaking the 'split-pane' components **CSS** so that the **detail** view is hidden on **mobile**
- Add a '**Proxy**' navigation service to simplify working with up to two NavControllers
- Somehow 'mark' your pages as **master** or **detail** so that we can tell in code what **type** of page we are dealing with
- Add the **split-pane** component to our app and listen to when its **activated/deactivated**
- **Wire it** all **together**

## 1. Tweak 'split-pane' component styles (LESS)

```
.split-pane-side:not(ion-menu) {
    display: initial;
}

.split-pane-main {
    display: none;
}

.split-pane-visible {
    .split-pane-main {
        display: block;
    }
}
```

## 2. Add a 'Proxy' navigation service

```
import { Injectable } from '@angular/core';
import { Nav } from 'ionic-angular';
import { PlaceholderPage } from
'../pages/placeholder/placeholder';
import { _DetailPage } from '../pages/_DetailPage';

@Injectable()
export class NavProxyService {

    _masterNav: Nav = null;
    get masterNav(): Nav {
        return this._masterNav;
    }
    set masterNav(value: Nav) {
        this._masterNav = value;
    }

    _detailNav: Nav = null;
    get detailNav(): Nav {
        return this._detailNav;
    }
    set detailNav(value: Nav) {
        this._detailNav = value;
    }

    _isOn: boolean = false;
    get isOn(): boolean {
        return this._isOn;
    }
    set isOn(value: boolean) {
        this._isOn = value;
    }
    pushDetail(page: any, params: any) {
        (this.isOn) ?
            this.detailNav.setRoot(page, params):
            this.masterNav.push(page, params);
    }

    pushMaster(page: any, params: any) {
        this.masterNav.push(page, params);
    }

    onSplitPaneChanged(isOn) {
        // set local 'isOn' flag...
        this.isOn = isOn;
        // if the nav controllers have been
instantiated...
        if (this.masterNav && this.detailNav) {
            (isOn) ? this.activateSplitView() :
                     this.deactivateSplitView();
        }
    }

    activateSplitView() {
        let currentView = this.masterNav.getActive();
            if (currentView.component.prototype
                instanceof _DetailPage) {
                // if the current view is a 'Detail'
page...
                // - remove it from the 'master' nav
stack...
                this.masterNav.pop();
                // - and add it to the 'detail' nav
stack...
                this.detailNav.setRoot(
                    currentView.component,
                    currentView.data);
            }
     }

    deactivateSplitView() {
        let detailView = this.detailNav.getActive();
        this.detailNav.setRoot(PlaceholderPage);
        if (detailView.component.prototype instanceof
_DetailPage) {
            // if the current detail view is a
'Detail' page...
            // ...so, not the placeholder page:
            let index =
this.masterNav.getViews().length;
            // add it to the master view...
            this.masterNav.insert(index,
                detailView.component,
                detailView.data);
        }
    }
}
```

Then add this to the 'providers' array of your angular module:

```
...
import {
    NavProxyService
} from '../services/NavProxy.service';
...

@NgModule({
    ...
    providers: [
        ...
        NavProxyService,
        ...
    ]
})
export class AppModule { }
```

## 3. 'Mark' pages as either 'master' or 'detail'

There are a number of ways you could achieve this…I opted to 'extend' Ionic page components with two custom abstract classes:

```
export abstract class _MasterPage { }
export abstract class _DetailPage { }
```

The **master** page, **Items** was then extended with `_MasterPage` , and the detail page, with `_DetailPage` :

```
...
@IonicPage()
@Component({
    ...
})
export class ItemsPage extends _MasterPage { ... }

...
@IonicPage()
@Component({
    ...
})
export class ItemPage extends _DetailPage { ... }
```

These are used by our **NavProxyService** to detect what type of page it is working with, and internally is used to decide which NavController to push a page to.

## 4. Add the 'split-pane' component to our app & listen for when it's activated/deactivated

In `/src/app/app.html` we're going to add the `ion-split-pane` component and two `ion-nav` components to handle our **master** and **detail** navigation stack (notice `(ionChange='…')` on the `ion-split-pane` ):

```
...
<ion-split-pane

(ionChange)="navProxy.onSplitPaneChanged($event._visib
le)">
    <ion-nav [root]="masterPage"
             #masterNav>
    </ion-nav>
    <ion-nav [root]="detailPage"
             #detailNav main>
    </ion-nav>
</ion-split-pane>
...
```

## 5. Wire it all together

This part is pretty simple, just set up our `NavProxyService` in `/src/app/app.component.ts'` :

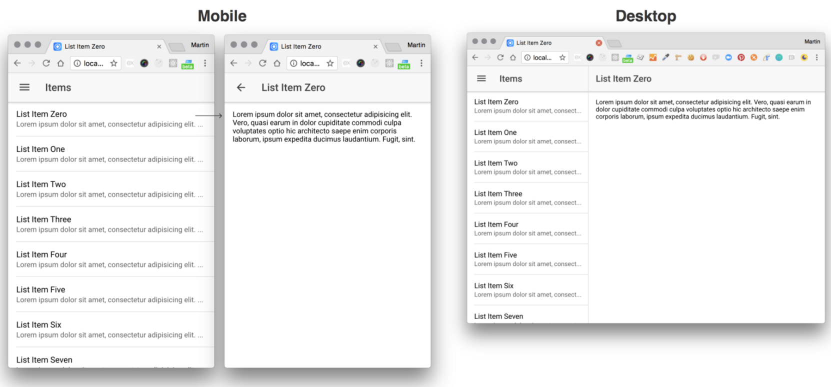
```
import { Component, ViewChild } from '@angular/core';
...
import { NavProxyService } from
'../services/NavProxy.service';
import { ItemsPage } from '../pages/items/items';
import { PlaceholderPage } from
'../pages/placeholder/placeholder';

@Component({
    ...
})
export class MyApp {
    // Grab References to our 2 NavControllers...
    @ViewChild('detailNav') detailNav: Nav;
    @ViewChild('masterNav') masterNav: Nav;

    ...

    constructor(
        ...
        private navProxy: NavProxyService) {
        platform.ready().then(() => {
            ...
            // Add our nav controllers to
            // the nav proxy service...
            navProxy.masterNav = this.masterNav;
            navProxy.detailNav = this.detailNav;
            // set initial pages for
            // our nav controllers...
            this.masterNav.setRoot(ItemsPage,
                { detailNavCtrl: this.detailNav });
            this.detailNav.setRoot(PlaceholderPage);
        });
    }

}
```

…and, amend `/src/pages/items` to use our `NavProxyService` d when requesting **detail** pages:

```
...
import { NavProxyService } from
'../../services/NavProxy.service';
import {_MasterPage, ItemPage } from '../';

@IonicPage()
@Component({
    ...
})
export class ItemsPage extends _MasterPage {

    constructor(public navCtrl: NavController,
                public navParams: NavParams,
                private navProxy: NavProxyService) {
        super();
    }

    onItemSelected(item) {
        // Rather than using:
        //     this.navCtrl.push(...)
        // Use our proxy:
        this.navProxy.pushDetail(ItemPage, item);
    }
```

```
  }
```

And that’s pretty much it. Just fire it up with an `ionic serve` and you should be away…



Lorem ipsum dolor sit amet, consectetur adipisicing elit. Vero, quasi earum in dolor cupiditate commodi culpa voluptates optio hic architecto saepe enim corporis laborum, ipsum expedita ducimus laudantium. Fugit, sint.

Hope that helps.

Demo code: [https://github.com/martinpritchardelevate/ionic-split-pane-demo](https://github.com/martinpritchardelevate/ionic-split-pane-demo)