Ember 2

Section 6 - Routing

Router Lifecycle
Defining a Route
Setting the Route's Model
Rendering a Template with a Route
Using a Route to Redirect

Section Objectives

- Understand the Ember route lifecycle
- Be able to define a route in the router
- Understand the Router's role in the Ember architecture
- Understand the Route's (Route Handler's) role in the Ember architecture
- Be able to set the route's model
- Be able to render a template with a route
- Be able to redirect from a route

Routing

Router Lifecycle

Router Lifecycle
Defining a Route
Setting the Route's Model
Rendering a Template with a Route
Using a Route to Redirect

Route Handler Lifecycle

- Called once for each route

Method	Description
beforeModel	Typically used if you need to do a data pre-check or you want to redirect immediately
model	Fetches and returns data that the Router can use to populate the controller's model
afterModel	Called after the model has been fetched but before the route has been setup

Route Handler Lifecycle

- Called each time the route's state changes

Method	Description
activate	Used to perform any special logic before route entry
setupController	Is used to setup any controller dependencies and populate the controller with data
renderTemplate	If you need to render additional or alternative templates, override this method
deactivate	Called during normal route exit to tear down any route-specific views

- **redirect()** – need to transition to a different route immediately

Routing

Defining a Route

Router Lifecycle
Defining a Route
Setting the Route's Model
Rendering a Template with a Route
Using a Route to Redirect

Defining a Route

- ***ember g route note***

- Generating a route with Ember CLI:
  - Creates a route entry in the Router
  - Creates a Template
  - Creates a Unit Test
  - Creates a Route

- If you need to add behaviors to your route, where would you add them?

- Can I add model data to the template through the Controller? Should you?

Basic Routes

- General route

router.js

```
Router.map(function() {
    this.route('about');
}); // /about
```

- Route with specifier

router.js

```
Router.map(function() {
    this.route('favorites', { path: '/favs');
}); // /favs
```

Nested Routes

- With nested routes, the nested route is rendered in the {{outlet}} of its parent; unless you break the chain. (Don't delete the application template!)

router.js

```
Router.map(function() {
    this.route('notes', function() {
        this.route('note');
    });
}); // /notes/note
```

- The application route is part of every application so you don't need to specify it in the router

Index Routes

- At every level of nesting, Ember automatically provides an index route (‘/’ path named index).

router.js

```
Router.map(function() {
    this.route(‘about’);
}); // /about
```

…is equal to…

router.js

```
Router.map(function() {
    this.route(‘index’, { path: ‘/’ });
    this.route(‘about’);
}); // /about
```

- The index template is rendered into the {{outlet}} in the application. If the user navigates to /about, Ember will replace the ‘index’ template with the ‘about’ template.

Dynamic Segments

router.js

```
Router.map(function() {
    this.route('note', { path: '/note/:note_id' });
}); // /note/2
```

- The **':note_id'** is an Ember convention, ***'model-name_id'.***

- It's this way for two reasons:
  1. They already know how to get to the model by default
  2. 'params' is an object and can only have one value per key

Routing

Setting the Route's Model

Router Lifecycle
Defining a Route
Setting the Route's Model
Rendering a Template with a Route
Using a Route to Redirect

The Route's Model

- JSON, Promise or Ember record returned from the model() of the Route Handler

application.js

```
model() {
    var notes = this.get('store').peekAll('note');
    return {
        notes: notes,
        singleNote: Ember.computed('notes.length', function() {
            return this.notes.get('length') == 1 ? true : false;
        }),
        selectedNote: null,
        editMode: false,
        numSelected: 0
    };
}
```

- Access model through Template **{{model.singleNote}}**

Dynamic Models

- this.route('note', {path: '/note/:note_id'});

note.js

```
model(params) {
    return this.get('store').peekRecord('note', params.note_id);
}
```

- A route with a dynamic segment will always have the model() executed if done through the URL; however, if done through a transition like the {{link-to}} helper, it does not if only the model context is provided. (e.g. note rather than note.id)

Multiple Models

- The RSVP.hash() takes parameters that return Promises
  - the RSVP.hash() does not return until all of the promises have resolved

js

```
model() {
    return RSVP.hash({
        notes:  this.get('store').findAll('note'),
        todos: this.get(‘store’).findAll(‘todo’)
    });
}
```

Routing

Rendering a Template with a Route

Router Lifecycle
Defining a Route
Setting the Route's Model
Rendering a Template with a Route
Using a Route to Redirect

Rendering a Route

- Each route is rendered through its associative template

router.js

```
Router.map(function() {
  this.route('note', function() {
    this.route('category', { path: ':/category/category_id');
  });
});
```

- A route is rendered in its parent's {{outlet}}

Custom Template

```
export default Ember.Route.extend({
    templateName: 'contacts/favorites'
});
```

js

- If you want finer control over the rendering of the template, you can use the renderTemplate() hook to specify the controller to use to configure the template and specific outlet to render it to

```
export default Ember.Route.extend({
    renderTemplate(controller, model) {
        let favorites = this.controllerFor('favoritePost');
        this.render('favoritePost', {
            outlet: 'posts',
            controller: favorites
        });
     }
});
```

js

Routing

Using a Route to Redirect

Router Lifecycle
Defining a Route
Setting the Route's Model
Rendering a Template with a Route
Using a Route to Redirect

Redirect Methods

- **transitionTo()**
  - works like {{link-to}}
  - appends the new route to the history
  - called from a Route Handler or *transitionToRoute()* from a Controller
- **replaceWith()**
  - works like {{link-to}}
  - replaces current route in the history
  - called from a Route Handler or *replaceRoute()* from a Controller

- If the new route has a dynamic segment, you need to pass either the model or the id of the segment. If you pass the model, the model() will not be called.

When To Transition

- Transition in the **beforeModel()** if you aren't dependent on the model
- Transition in the **afterModel()** if you are dependent on the model
- If you redirect in the **afterModel()** and you are transitioning to a child route, the **beforeModel()** and **afterModel()** will be called in the parent route again
  - Prevent this by transitioning in the **redirect()**

```
redirect(model, transition) {
    if (model.get('length') === 1) {
        this.transitionTo('posts.post', model.get('firstObject'));
    }
}
```

Lab 5 - Routes

- In our Notes app, we have been given new requirements that we need to provide folders for our notes. The user will already have access to the default folder, ‘All notes’. They are allowed to create new ones. Create a ’notes’ route that will contain the folders for all of the notes. When they select a folder, replace the master view with the notes in that folder and a back button to go back.
  - You will need to create two routes:
    1. Route to display all folders in the master view (e.g. /notes)
    2. Route to display only notes in selected folder (e.g. /notes/ember)
  - Reuse ‘note-row’ template markup and fit to folders for now. We will create a component for it in the next section.
  - Use a computed property to get the selected note in the application route

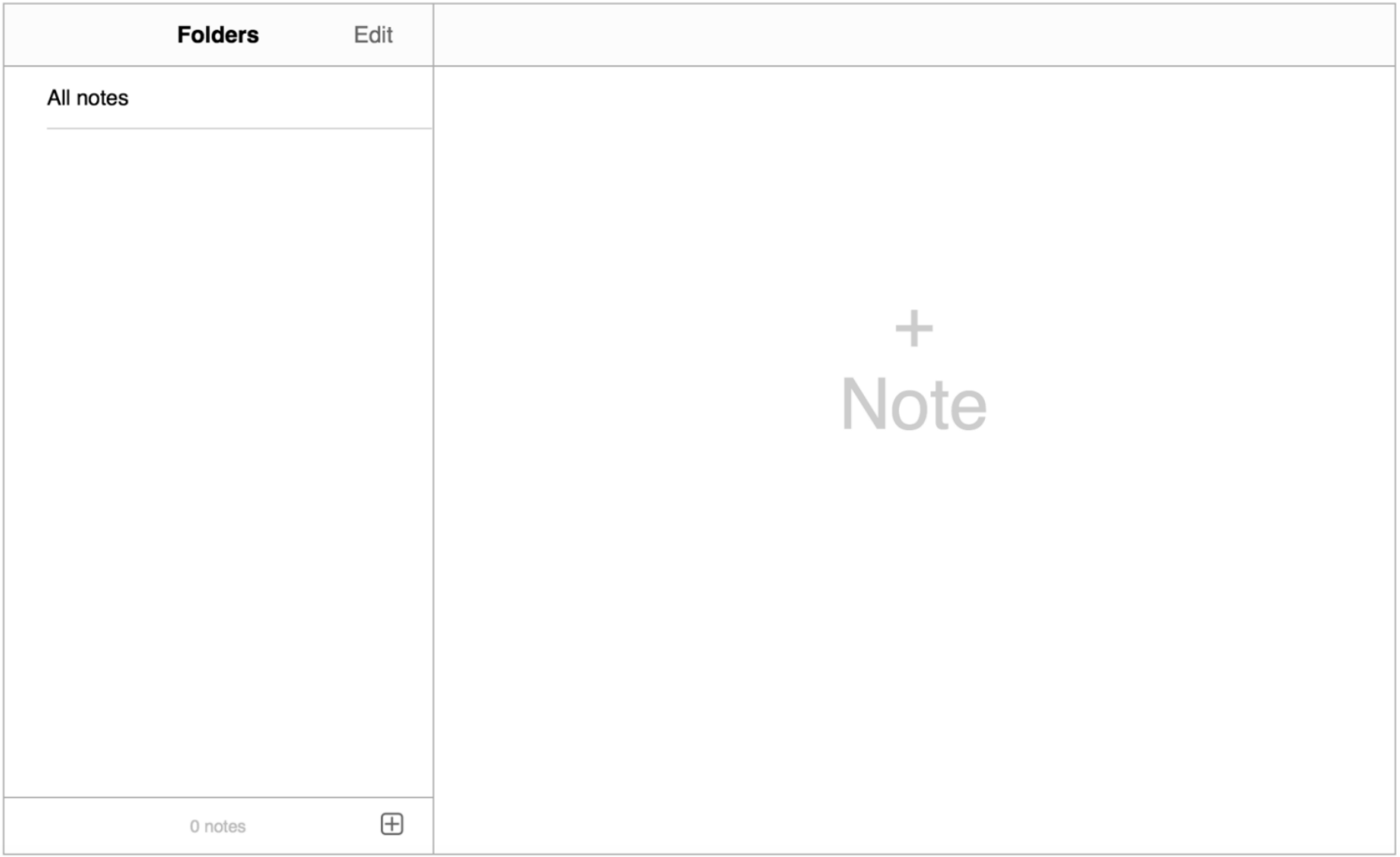
Lab 5 - Routes

- Use this space to draw out your design.

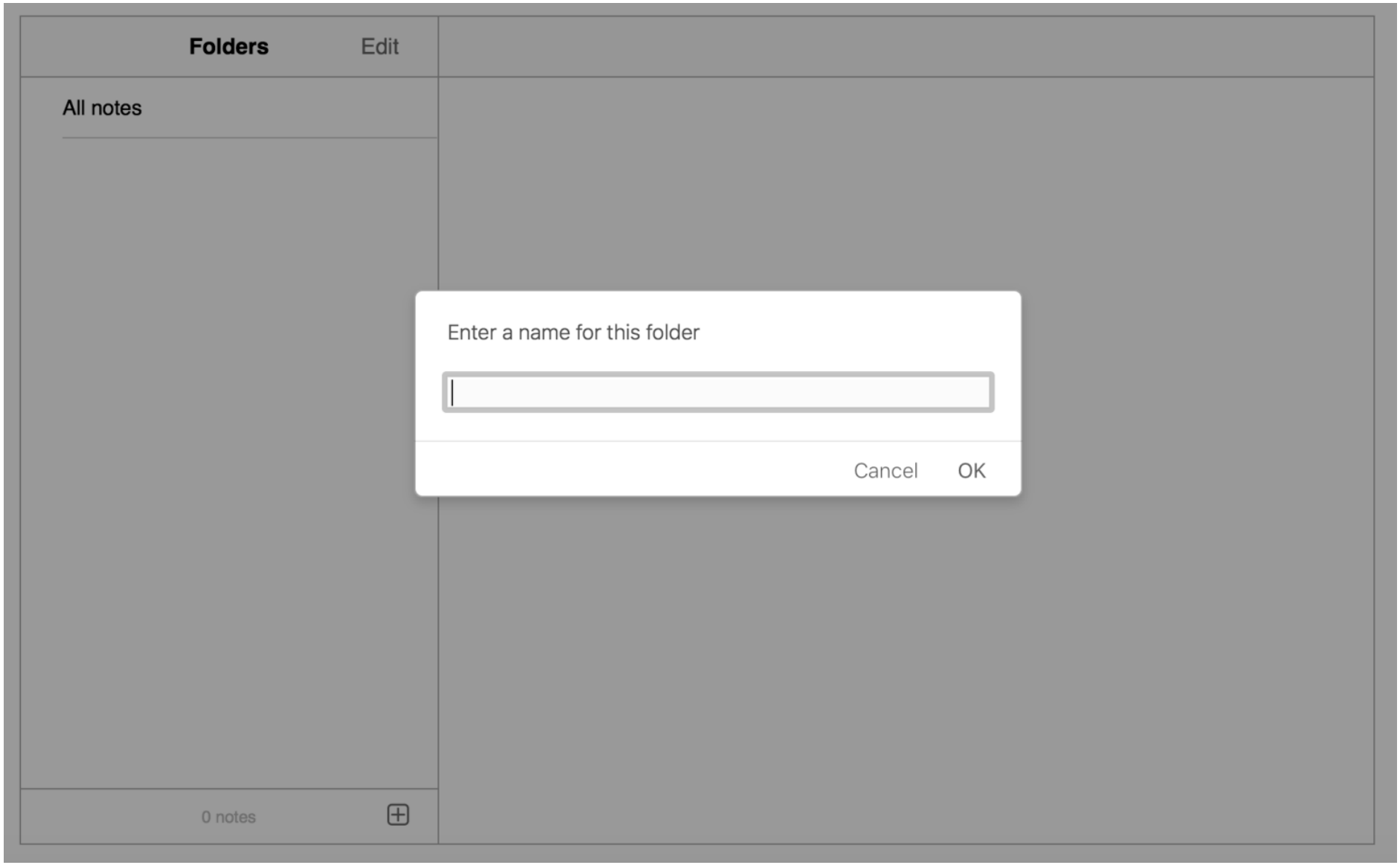
Section Review

- Understand the Ember route lifecycle
- Be able to define a route in the router
- Understand the Router's role in the Ember architecture
- Understand the Route's (Route Handler's) role in the Ember architecture
- Be able to set the route's model
- Be able to render a template with a route
- Be able to redirect from a route

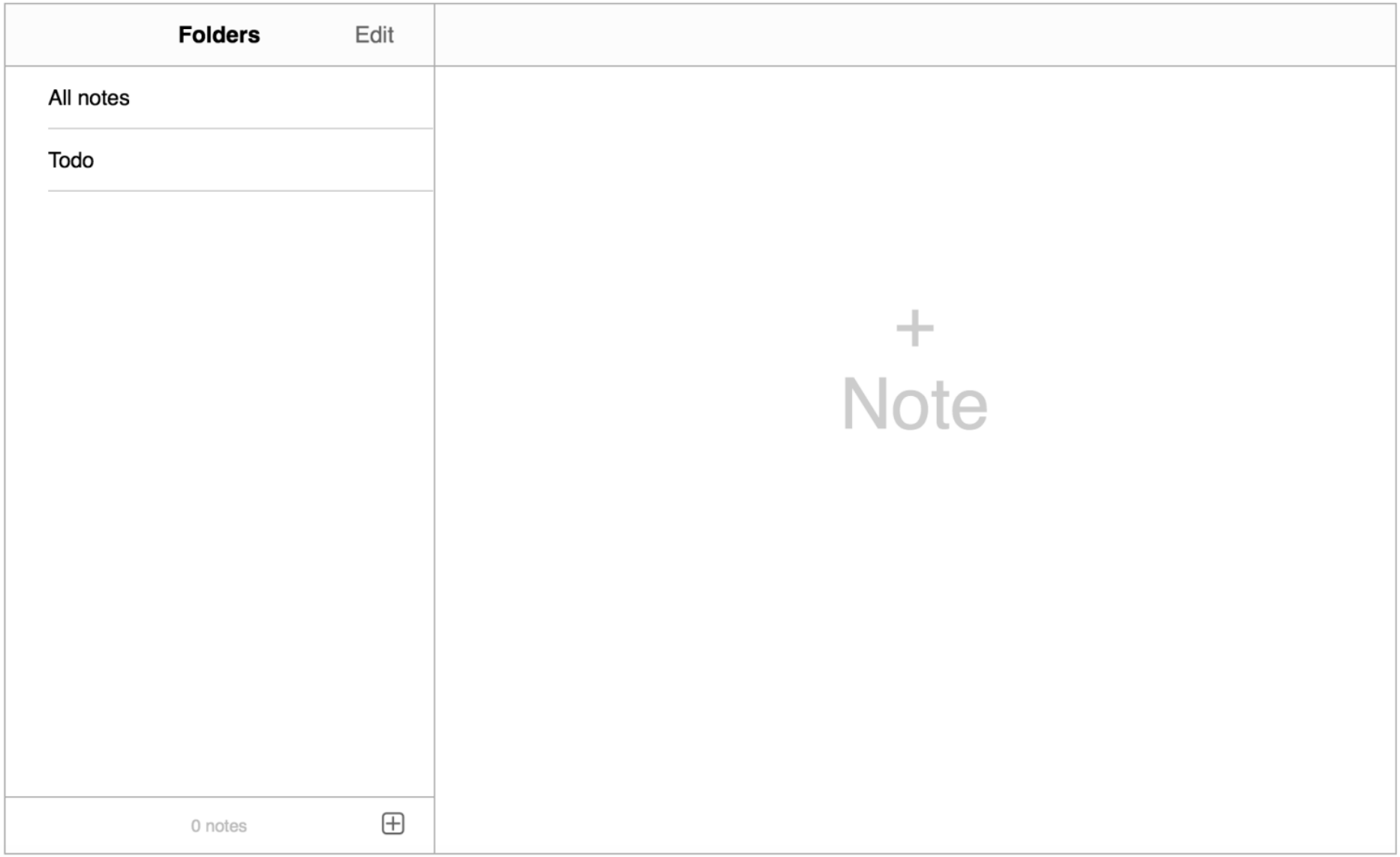
Screenshot – Home Page



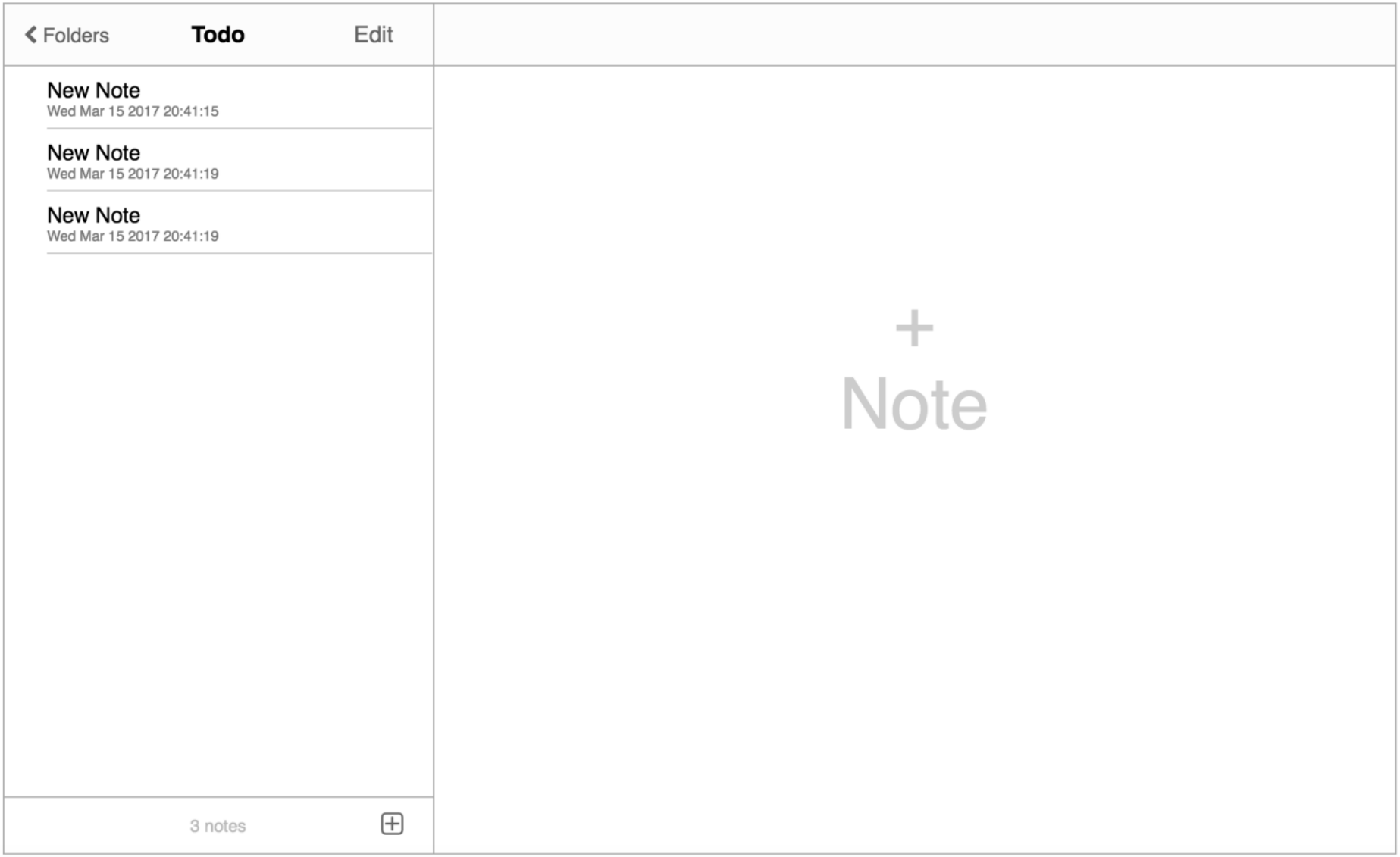
Screenshots – Add Folder



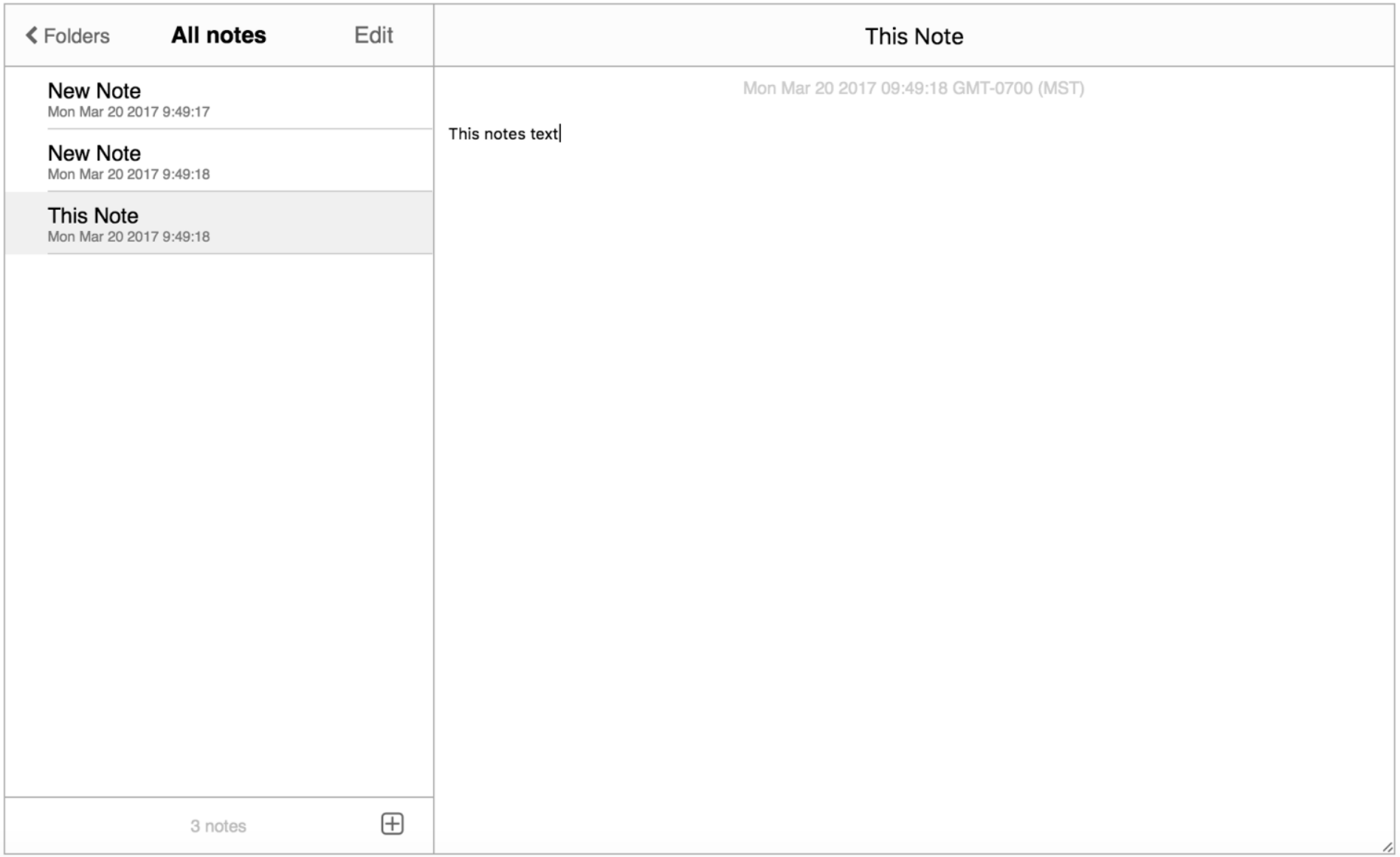
Screenshots – Folder Added



Screenshots – Add Note



Screenshots – Selected Note



End of Section

- This slide is purposely devoid of any information