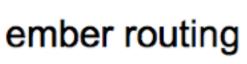
Google







All Videos More Settings Tools Shopping Images News

About 286,000 results (0.56 seconds)

Routing - Ember Guides - Ember.js

https://guides.emberjs.com/v1.10.0/routing/ ▼

A description for this result is not available because of this site's robots.txt Learn more

Ember.js - Ember.Route

https://emberjs.com/api/classes/Ember.Route.html ▼

Route Class packages/ember-routing/lib/system/route.js:72 ... Route class is used to define individual routes. Refer to the routing guide for documentation.

Ember Routing: The When and Why of Nesting | Hashrocket

https://hashrocket.com/blog/posts/ember-routing-the-when-and-why-of-nesting ▼ by Andy Borsz on July 29, 2013. Understanding the proper use of the Router in an Ember.js app is a fundamental core concept of the framework. The docs tell us ...

Ember.js Routing and Rendering Simplified - Matthew Lehner

matthewlehner.net/ember-js-routing-and-views/ ▼

v1.10.0

GETTING STARTED

GETTING EMBER

CONCEPTS

THE OBJECT MODEL

APPLICATION

TEMPLATES

ROUTING

Introduction

Defining Your Routes

Generated Objects

Specifying a Route's Model

Setting Up a Controller

Rendering a Template

Redirecting

Specifying the URL Type

Query Parameters

Asynchronous Routing

nading / Error Substates



INTRODUCTION



ROUTING

As users interact with your application, it moves through many different states. Ember.js gives you helpful tools for managing that state in a way that scales with your application.

To understand why this is important, imagine we are writing a web app for managing a blog. At any given time, we should be able to answer questions like: Is the user currently logged in? Are they an admin user? What post are they looking at? Is the settings screen open? Are they editing the current post?

In Ember.js, each of the possible states in your application is represented by a URL. Because all of the questions we asked above— *Are we logged in? What post are we looking at?*—are encapsulated by route handlers for the URLs, answering them is both simple and accurate.

At any given time, your application has one or more active route handlers. The active handlers can change for one of two reasons:

- 1. The user interacted with a view, which generated an event that caused the URL to change.
- 2. The user changed the URL manually (e.g., via the back button), or the page was loaded for the first time.

When the current URL changes, the newly active route handlers may do one or more of the following:

- 1. Conditionally redirect to a new URL.
- 2. Update a controller so that it represents a particular model.