Ember.js Cheat Sheet

A quick reference guide for Ember.js, covering components, routing, data management, and testing.

Ember Essentials

Application Setup

- `ember new my-app` - Create a new Ember application.
- `ember serve` or `ember s` - Start the development server.
- `ember build` or `ember b` - Build the application for production.
- `ember g <blueprint> <name>` - Generate code using blueprints (e.g., `ember g component my-component`).
- `ember test` or `ember t` - Run tests.
- `ember destroy <blueprint> <name>` - Destroy generated code (e.g., `ember destroy route about`).

Basic Concepts

Components	Reusable UI elements. Defined with `.js` (logic) and `.hbs` (template) files.
Routes	Manage application state and URL transitions. Define models and actions.
Templates	Handlebars templates for rendering UI.
Models	Represent data. Often fetched from an API using Ember Data or custom solutions.
Services	Singletons for managing global application state or logic.
Ember CLI	The command-line interface for managing Ember projects.

Components

Component Definition

Component JavaScript file (`app/components/my-component.js`):

```
import Component from '@glimmer/component';

export default class MyComponent extends Component {
  // Component logic here
}
```

Component template file (`app/components/my-component.hbs`):

```
<div>
  <h1>Hello, {{this.args.name}}!</h1>
</div>
```

Component Arguments

Passing Arguments	<MyComponent @name="World" />
Accessing Arguments	Use `this.args.name` in the component template or `this.args` in the component's JavaScript file.

Component Actions

Defining Actions

```
import Component from '@glimmer/component';
import { action } from '@ember/object';

export default class MyComponent extends Component {
  @action
  myAction() {
    // Action logic here
  }
}
```

Invoking Actions in Template	<button {{on "click" this.myAction}}>Click Me

Routing

Route Definition

Route JavaScript file (`app/routes/my-route.js`):

```
import Route from '@ember/routing/route';

export default class MyRoute extends Route {
  model() {
    // Fetch data here
  }
}
```

Route template file (`app/templates/my-route.hbs`):

```
<h1>My Route</h1>
```

Router Configuration

Configure routes in `config/router.js`:

```
import EmberRouter from '@ember/routing/router';
import config from 'my-app/config/environment';

export default class Router extends EmberRouter {
  location = config.locationType;
  rootURL = config.rootURL;
}

Router.map(function() {
  this.route('about'); // Defines the /about route
  this.route('posts', function() { // Defines the /posts route with nested routes
    this.route('new');
    this.route('show', { path: '/:post_id' });
  });
});
```

Linking Between Routes

Using the `{{link-to}}` Helper	{{#link-to "about"}}About{{/link-to}}
Passing Model Data	{{#link-to "posts.show" post}}View Post{{/link-to}}

Data Management

Ember Data (Basic)

Defining a model (`app/models/post.js`):

```
import Model, { attr } from '@ember-
data/model';

export default class PostModel extends Model {
  @attr('string') title;
  @attr('string') body;
}
```

Fetching data in a route:

```
import Route from '@ember/routing/route';
import { inject as service } from
'@ember/service';

export default class PostsRoute extends Route
{
  @service store;

  model() {
    return this.store.findAll('post');
  }
}
```

Computed Properties

```
import { computed } from '@ember/object';
import Component from '@glimmer/component';

export default class MyComponent extends
Component {
  @computed('firstName', 'lastName')
  get fullName() {
    return `${this.firstName}
${this.lastName}`;
  }
}
```

Services

Creating a service (`app/services/my-service.js`):

```
import Service from '@ember/service';

export default class MyService extends Service
{
  property = 'default value';

  myMethod() {
    // Service logic here
  }
}
```

Injecting a service:

```
import Component from '@glimmer/component';
import { inject as service } from
'@ember/service';

export default class MyComponent extends
Component {
  @service myService;

  // Use this.myService to access the service
}
```