Gmail **Amy Lam <amy.r.lam@gmail.com>**

[PREVIEW] The Ember Times - Issue No. 158
1 message

embertimes <amy.r.lam@gmail.com> Sat, Aug 1, 2020 at 1:03 PM
To: embertimes <amy.r.lam@gmail.com>

👋 Emberistas! 🐹

Ember 3.20 Released 🚀, a series of blog posts on ember-modifer and its internals 🎉, detect when Ember components enter or leave the viewport 🔍, a blog post and RFC on Ember.Component 📖, and last, but not least, document Ember apps with Docfy 📗!

Ember 3.20 Released 🚀

A new Ember.js blog post is up to announce the release of version 3.20 of Ember.js, Ember Data and Ember CLI!

3.20 includes several new features for the ecosystem.

- In Ember.js, the `{{in-element}}` helper is available as public API. This helper solves rendering challenges similarly to `ember-wormhole` and `ember-elsewhere`. It allows rendering content into a destination elsewhere on a page.
- Ember Data explicitly supports the combined use of `EmbeddedRecordsMixin` and `JSONAPISerializer` through the `isEmbeddedRecordsMixinCompatible` property.
- Ember CLI allows syncing Blueprints when running `npx ember-cli-update`, which avoids some potential issues that previously existed when running this flow. Check out the related RFC to learn more.

In Ember.js, `Meta.prototype.setSourceDestroyed` and `Meta.prototype.setSourceDestroying` are now deprecated. There were no deprecations for Ember Data. Ember CLI issues a warning for using Node 13 and deprecates the use of the `PACKAGER` experiment.

For more information, give the Ember.js blog post a read.

A series of blog posts on ember-modifer and its internals 🎉

Have you been looking to get started with ember-modifier in your Ember.js applications?

Raja SK (@RajaSK05) wrote a series of blog posts on **Ember modifiers**.

How do Ember Modifiers get to be managed internally? 🥼

In continuation of his first blog post reusable DOM behavior in React vs Ember, Raja wrote a second post detailing the ember-modifier internals and what an **element modifier manager** is. He begins by talking about the modifier lifecycle methods. A modifier lifecycle consists of:

- createModifier()
- installModifier()
- updateModifier()
- destroyModifier()

In the process of explaining the lifecycle methods, Raja details each step with code snippets that help the reader understand what each of them means.

Read more about the element modifier manager and its internals on dev.to today!

The magic behind Ember modifiers ✨

Raja SK's third blog post on Ember modifiers is titled the magic behind the ember modifiers, where he talks about the internal workings of an Ember modifier - a feature that is offered by Ember Octane.

In the process of explaining how a modifier works, he showcases its use by writing a simple autofocus modifier using the functional modifier approach.

For more details on the blog, check it out on dev.to.

Demystifying ember-render-modifiers 😎

Finally, Raja SK's fourth blog post talks about demystifying ember-render-modifiers.

He talks about ember-render-modifiers, which provide element modifiers that can be used to hook into specific portions of the rendering lifecycle of a component.

There are several addons based on ember-modifier, but **ember-render-modifiers** makes it easy to understand its implementation. If you are looking to write your own custom modifiers, then ember-render-modifiers is a good place to start with. Read more on the blog post on dev.to today!

P.S. Robert Jackson (@rwjblue) recently published a major version bump of ember-modifier to v2.x, check out the Changelog!

Detecting Ember.js components entering or leaving the viewport 🔍

Koushik Radhakrishnan (@Koushikrad) wrote a blog post on detecting when Ember components are entering or leaving the viewport.

Consider a dashboard with 10 widget components, each of which makes an API request. When a user lands on the dashboard page, do we need all the widgets to fetch their data at the same time, even if the user's viewport shows only 5 widgets at a time?

A solution is to use the ember-in-viewport addon, which detects if an Ember component has entered the browser's viewport. The addon tries a few different approaches (`IntersectionObserver` API, then `requestAnimationFrame`, then the Ember run loop and event listeners) to detect if a DOM element is in the user's browser.

By hooking up with the addon’s provided `inViewport` service, we are able to request data once the component is within the viewport.

```
const { onEnter } = this.inViewport.watchElement(this.element, { viewportTolera
[illegible]
```

```
onEnter(this._renderInView.bind(this));
```

The components that are not in view will not make a request for API data, unlike the route’s model hook using a `Promise.all`.

Read more about detecting components from the blog post!

Blog post and RFC on Ember.Component 📖

There's an interesting perspective about how your team might handle usage of Ember's built-in components in a `GlimmerComponent` world presented in Mehul Kar's (@mehulkar) recent blog post.

The post indicates that in order to avoid introducing breaking changes you may want to consider not extending `Ember.Component` since `GlimmerComponent` does not have the same methods to support such extension.

This blog post continues the conversation Mehul started in an RFC on the subject earlier this year that proposes making `@ember/component` an optional feature. Conversations like these are vital to the health of Ember's ecosystem! So if you're interested in participating in this conversation and have opinions, head on over to the RFC and provide some feedback.

Document Ember apps with Docfy 📗

Hope you didn't miss an exciting announcement earlier in July. Josemar Luedke (@josemarluedke) released Docfy, a cross-framework tool to help you build documentation sites from Markdown files.

The best part? Docfy provides official support for Ember.js! You can style the Docfy components and leverage existing remark plugins too.

Check out Getting Started to learn more about adding Docfy to your Ember apps.

Contributors' corner 👏

This week we'd like to thank @abhilashlr, @Alonski, @arthirm, @bekzod, @cah-briangantzler, @chancancode, @chrisrng, @CodingItWrong, @dfreeman, @emonroy, @ijlee2, @IzzatN, @jaredgalanis, @jenweber, @kategengler, @kiwiupover, @locks, @loganrosen, @MelSumner, @NullVoxPopuli, @ppegusii, @pzuraq, @RichardOtvos, @rwjblue, @scalvert, @snewcomer, @SYU15 for their contributions to Ember and related repositories! 💖

Connect with us 🤓

Wondering about something related to Ember, Ember Data, Glimmer, or addons in the Ember ecosystem, but don't know where to ask? Readers' Questions are just for you!

Submit your own short and sweet **question** under bit.ly/ask-ember-core. And don't worry, there are no silly questions, we appreciate them all - promise! 🤞

Want to write for the Ember Times? Have a suggestion for next week's issue? Join us at #support-ember-times on the Ember Community Discord or ping us @embertimes on Twitter.

Keep on top of what's been going on in Emberland this week by subscribing to our e-mail newsletter! You can also find our posts on the Ember blog.

That's another wrap! ✨

Be kind,

Chris Ng, Abhilash LR, Isaac Lee, Jared Galanis, Amy Lam and the Learning Team