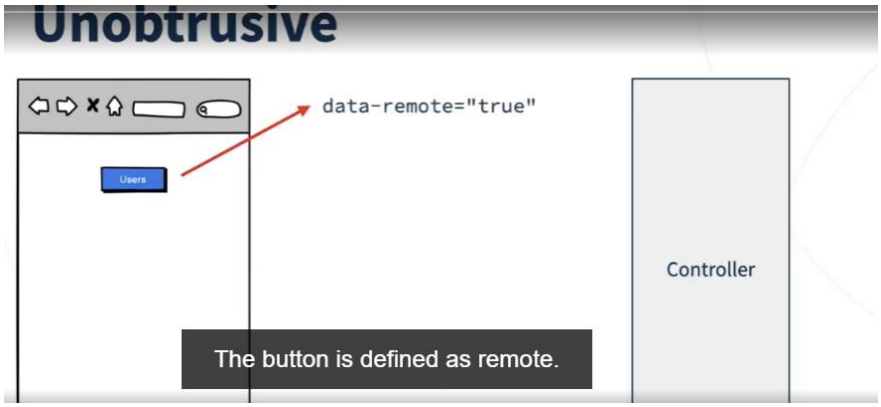
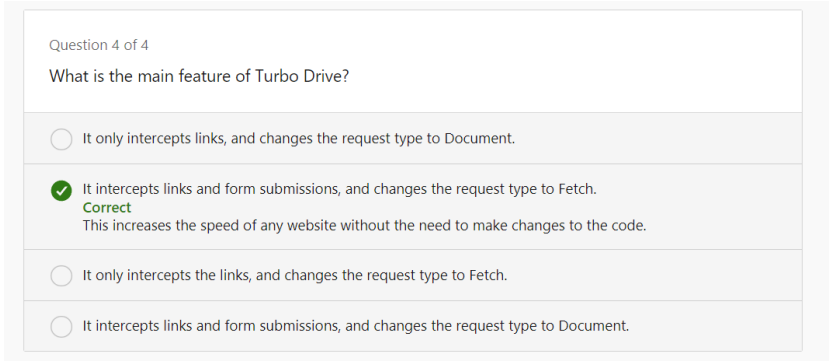
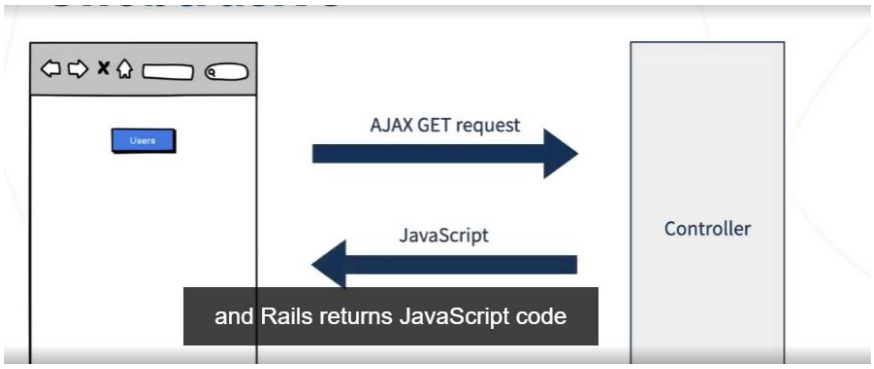
Hotwire In Rails

In nonreactive way all page is being reload. In reactive way, there are Turbolinks. Turbolinks intercepts all links automatically and modify requests. It converts the request to make it async with AJAX. Turbolinks extract the head to update the current one and replaces the body. Thus, only body is reloaded. But we need to go further. In order to update only the part of the body, we need to use JS.

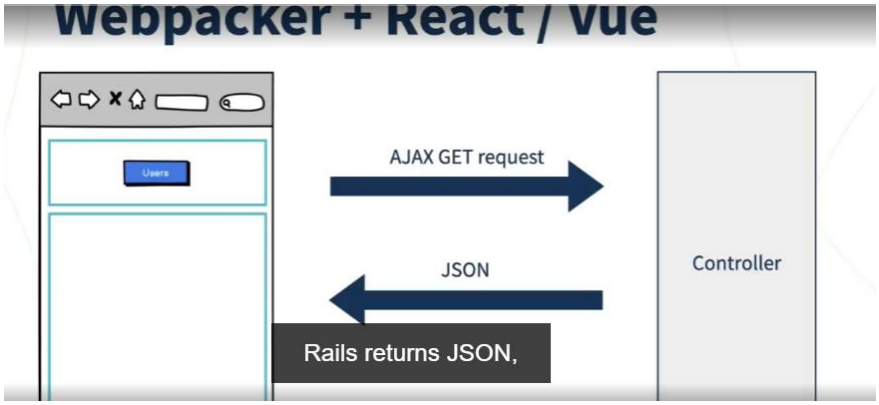


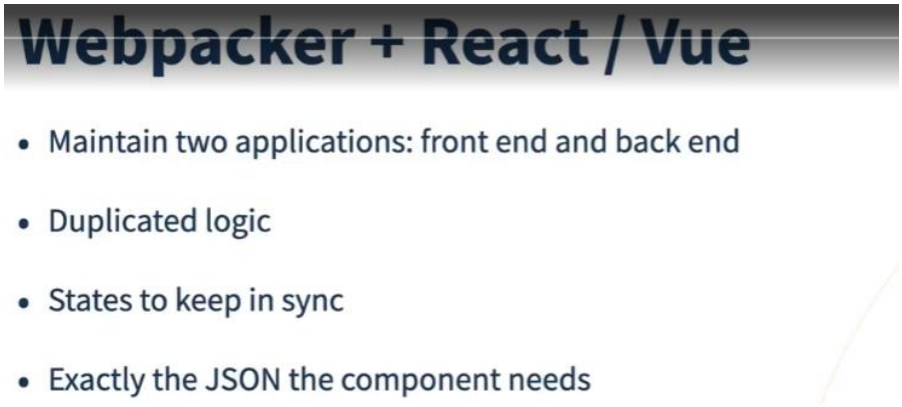


The button is defined as remote. This way AJAX GET request is sent and Rails return JS code that updates specific parts of the page.



Another good solution is using Webpacker and React/Vue. This way page elements would be grouped into components. After AJAX get request, Rails return JSON, which is transformed to DOM elements by front-end app and updates necessary components. This way adds compexity since we'd have Rails render JSON on one side and JS framework rendering components on the other.

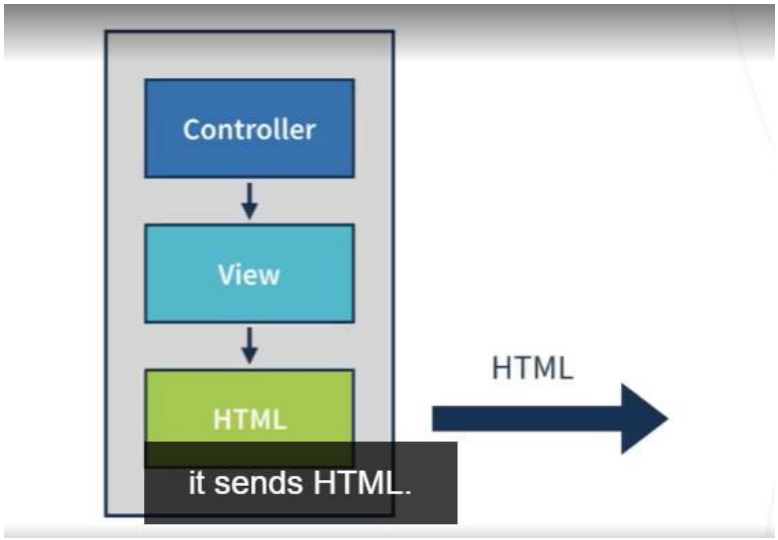




So, Hotwire offers a simpler way to make reactive apps.

Hotwire (HTML over Wire) renders HTML on the server in the traditional RoR way.

Instead of sending JSON over js, it sends HTML.



Hotwire consists of two parts:

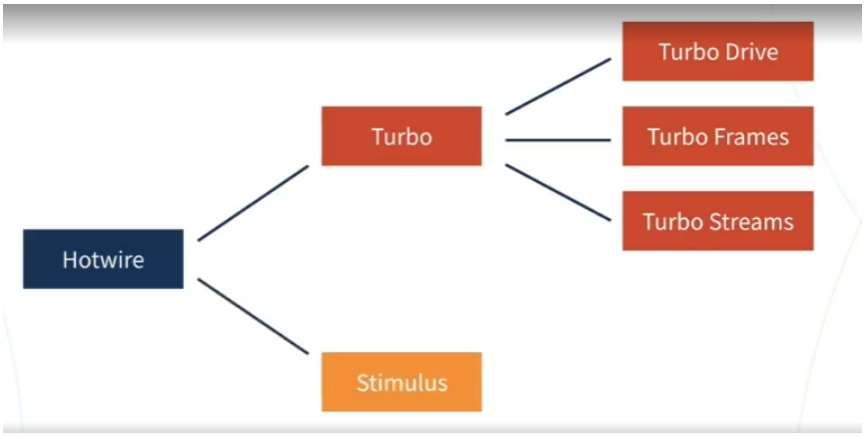
1. Turbo.

Updates parts of a page dynamically without having to write JS. Turbo consists of three parts:

- Turbo Drive

- Turbo Streams
- Turbo Frames

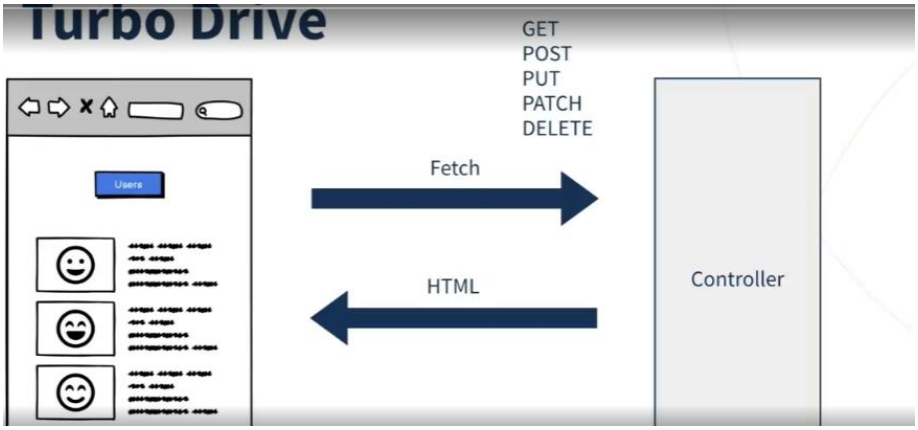
2. Stimulus.
   A JS framework that compliments Turbo.



Let's take a look at Turbo.

- **Turbo Drive**

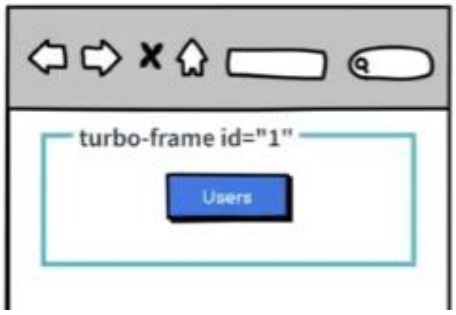
is a replacement for Turbolinks. It intercepts links automatically and modifies the request using Fetch API. When it receives the response, it separates the head from the body. Then it merges the head first and updates the body next. But instead of using just GET (as old turbolinks), Turbo Drive also intercepts other HTTP requests. This means that forms also work async automatically without reloading the whole page. The minus: the body is completely replaced.



If you need to update only the part of the page, use Tubo Frames.

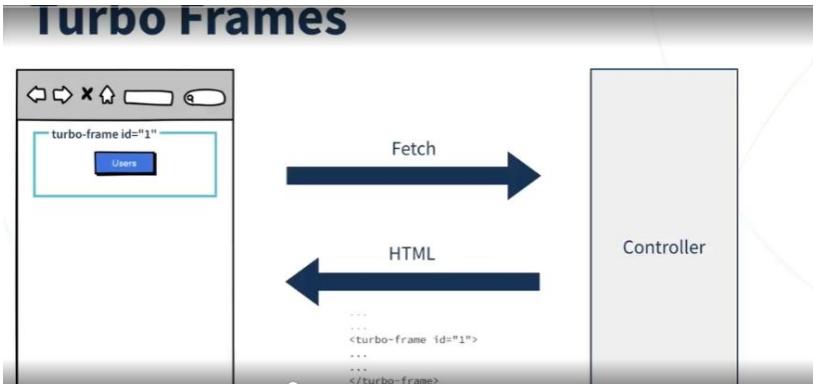
- **Turbo Frames**

Based on defining parts of the page as frames.
In our example the button area is defined as a turbo frame assigning it a unique identifier.



From now on, any request made by this frame will have its response inside. That is, the frame will be reloaded. So, when the button is clicked, the Fetch request is made and HTML response is returned. Turbo then extracts the frame with the same id from that HTML and updates it.

So, only part of the page was reloaded. Imagine different case.

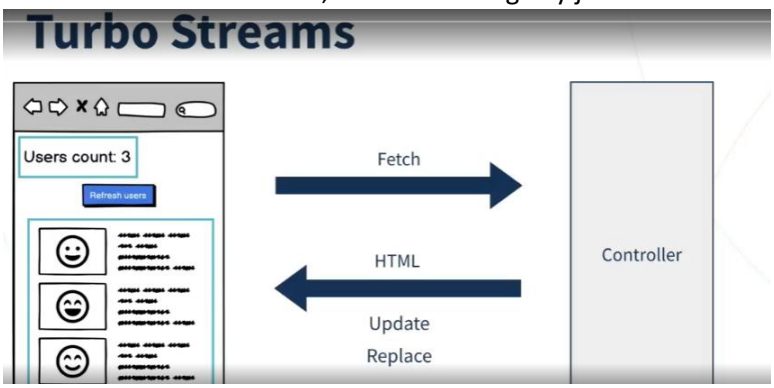


Let's look at **Turbo Streams**.

- If you should reload different components on the page, you should use Turbo Streams.

  When clicking the button, a fetch request is made and the controller returns HTML with streams. Turbo uses streams to perform actions on the page.

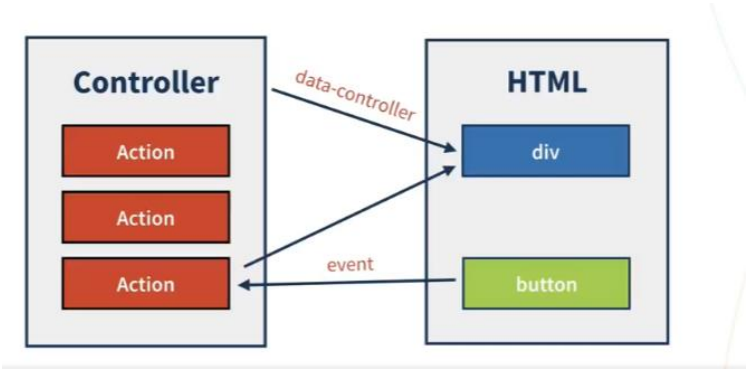
  In this case, an update stream for the counter of the page and a replace stream for the list. All in HTML format, without writing any js.



If you need to write JS, use Stimulus.

Stimulus organizes JS code in the form of controllers.

Using an HTML attribute, you connect an elements in the DOM to the controller. Now when event is triggered by another element, you can control it.



- **Hotwire** incrementally improves your Ruby on Rails apps
- **Turbo Drive** increases speed by intercepting requests automatically
- **Turbo Frames** reloads parts of the page by framing them
- **Turbo Streams** does actions on multiple parts of the same page

  Broadcasting via WebSockets with ActionCable
- **Stimulus** is included as a simple JavaScript framework

Turbo Drive acts automatically on all links and forms.

Sometimes you will need to not use it. To disable it. Like when working with gems that don't support it. For instance, Devise.

We can disable it globally or locally.

Globally:

```
import { Turbo } from "@hotwired/turbo-rails"
import "controllers"

Turbo.session.drive = false;
```

in application.js

Also,

```
<%= stylesheet_link_tag "tailwind", "inter-font", "data-turbo-track": "reload" %>

<%= stylesheet_link_tag "application", "data-turbo-track": "reload" %>
```

In application.html.erb means that Turbo will monitor all css in the app.

The main difference between turbo frames and turbo streams is that turbo streams allow you to update different parts of the page.

Turbo Streams allow not only to replace something but also add elements to the block (append), prepend, remove and etc.

Stimulus