Vite Vue Documentation

Release 0.1

synw

Jun 14, 2017

Install

CHAPTER 1

Install

```
pip install django-vitevue
```

Add to installed apps:

```
"vv",
```

Add to the bottom of urls.py:

```
urlpatterns.append(url(r'^',include('vv.urls')))
```

Include a vue block in the main template:

```
{% block vues %}{% endblock %}
```

Load the libraries in html header:

```
<script type="text/javascript" src="{% static 'js/vue.min.js' %}"></script>
<script type="text/javascript" src="{% static 'js/vuex.js' %}"></script>
<script type="text/javascript" src="{% static 'js/page.js' %}"></script>
<script type="text/javascript" src="{% static 'js/axios.min.js' %}"></script>
<script type="text/javascript" src="{% static 'vv//vv.js' %}"></script>
<script type="text/javascript" src="{% static 'vv/vvstore.js' %}"></script>
```

CHAPTER 2

Manage a frontend from a module

To make a module using Vuejs frontend structure your module's template folder should contain this:

```
vues/data.js
vues/methods.js
vues/computed.js
vues/extra.js
routes.js
```

Put your different vuejs parts at the appropriate places.

Note: *extra.js* is just extra global javascript, the rest are Vue parts to be assembled.

Templates

Put a `{% block vues %}{% endblock %}` in your base template

Settings

Declare your app in settings:

```
VV_APPS = ["my_app"]
```

Your frontend parts will be merged into the main app

Client side routing

Optional routing: make a `routes.js` file into your module template folder and fill it with page.js routes if needed: ex:

```
page('/someurl/', function(ctx, next) { app.doSomething() } );
```

CHAPTER 3

Available methods

State management

A basic state management mechanism is used to manage the display. Methods:

`flush()`: reset all active items: all the active variables will be reseted: a string will be set to `""`, a number to `0`, an object to `{}` and a boolean to `false`

To preserve an element from beeing flushed you can pass it to the function: `flush("item")`

`activate(["item1", "item2"])`: set active the given items

`isActive("item")`: return true or false, usefull for v-show

```
this.flush();
this.activate(["myitem"]);
```

Get and post data

`loadData(url, action)`: loads data and process actions on it

```
function error(err) {
   console.log(err)
}
function action(data) {
   console.log(data)
}

this.loadData("{% url 'myurl' %}", action, error);
```

`postForm(url, data, action, error, csrfmiddlewaretoken)`: post a form

Note: the `csrfmiddlewaretoken` is optional an will be set from the session cookie if not provided

Ex: in `mymodule/templates/vue/methods.js`:

```
postMyForm: function() {
   function error(err) {
   console.log(err)
   }
   function action(response) {
      console.log(response.data)
   }
   var form = document.getElementById("myform");
   var data = this.serializeForm(form);
   this.postForm(url, data, action, error, data.csrfmiddlewaretoken)
},
```

Pass the form token if posting a Django form, otherwise use the session cookie

`str(json_obj)`: shortcurt for pretty JSON.stringify

CHAPTER 4

Ressources

Applications

- Vvpages : pages management
- VVcontact : contact form
- VVphotos : mobile friendly photos albums
- VVcatalog : products catalog with cart
- VVlogin : inline login form