django-partitialajax

Feb 12, 2022

General:

1 Installation **3**
- 1.1 Only with Python/django . . . 3
- 1.2 JS / Python Setup . . . 3
- 1.3 General Setup . . . 4

2 Information **5**
- 2.1 Limitations . . . 5
- 2.2 Recommend . . . 5

3 Ajax JSON Protocol **7**

4 Template Tags **9**

5 How to use **11**

6 Template Tag (with automated js) **13**

7 Js Manual **15**
- 7.1 Options . . . 15

8 Mixin **17**

9 PartitialAjax **19**
- 9.1 Options . . . 20
- 9.2 Events . . . 20

10 Template Tags **21**

11 Indices and tables **23**

Python Module Index **25**

Index **27**

Last Version 0.2.2

CHAPTER 1

Installation

1.1 Only with Python/django

Install with:

```
pip3 install django-partitialajax
```

Than put partitialajax into your INSTALLED_APPS:

```
INSTALLED_APPS = [
...
    partitialajax
]
```

Add the JS libary to each page you have Partitials:

```
...
{% static 'partitialajax/index.js' %}
</head>
...
```

1.2 JS / Python Setup

You can also use the js src to build your own javascript set for every page

Install with

```
pip3 install django-partitialajax
npm install django-partitialajax --save
```

Tipp: If you use webpack for your js code use the django-webpack-loader libary.

To use the "autodiscover" for elements with partitial loading use the following JS code:

1.3 General Setup

You can define a partitial without a line of your own js code:

```
{% load partitialajax %}

{% direct_partitial ".content" %}
```

All options, see: *Options* can be set as element Attribute:

```
{% load partitialajax %}

{% direct_partitial ".content" url:"remotepath" %}
```

CHAPTER 2

Information

This libary can helps you to replace parts of your django project.

to make your content replacement as comfortable as possible the library offers you

- django: templatetags
- django: mixins
- js/npm: js libary

With these options its very easy to replace parts of your project.

2.1 Limitations

- This libary doesnt work with functional based views
- This libary doesnt handle submits from partitial forms (but you can just upgrade this)

2.2 Recommend

You can use webpack to build your own js files. just simply use *npm install django-partitialajax*

CHAPTER 3

Ajax JSON Protocol

type	*object*		
properties			
• **content**	type	*object*	
	properties		
	• key	type	*string*
	• value	type	*string*
• **options**	type	*object*	
	properties		
	• **reregister**	type	*boolean*
		default	True
definitions			

CHAPTER 4

Template Tags

- reload
- allowed_elements
- url
- (selectorstring)

CHAPTER 5

How to use

two ways of use

- use mix of templatetag and automatic js
- use manual js only

CHAPTER 6

Template Tag (with automated js)

Different Template Tags:

- Direct Partitial
- Lazy Partitial

Direct Partitial Recives content direct on Initial main page load Lazy Partitial Requests its own content only by trigger

Both Partititals have the following Features:

- Define Reload Button
- Define

CHAPTER 7

Js Manual

Initialize PartitialAjax Object. This object recives one option dict

```
// PartitialAjax(<options>)
```

The options dict: url: “”, element: “”, onlyChildReplace: true, interval: 5000, allowedElements: “all”, textEventCallback: “console.info”, restrictRemoteConfiguration: true, configFromElement: true

7.1 Options

7.1.1 url

Remote Partitial URL **default**: Current URL

CHAPTER 8

Mixin

class partitialajax.mixin.**CreatePartitialAjaxMixin**

class partitialajax.mixin.**DeletePartitialAjaxMixin**

class partitialajax.mixin.**DetailPartitialAjaxMixin**

class partitialajax.mixin.**ListPartitialAjaxMixin**

class partitialajax.mixin.**PartitialAjaxMixin**

Add this Mixin to your View to activate the useage of PartitialAjax

get_partitial(*origin*, *context*)

Collect Partitial Contents from template or remote url

Parameters

- **origin** – tuple or string; if tuple: the first value is the path to template or remote the second is the keyword "template", "remote"
- **context** – context used for rendering (remote paths also renderd with this context)

Returns

get_partitial_context(*context*)

Modifys content for ajax Partitials Detailed: It loads all defined files from partitials_list and render this templates

Parameters **context** – context dict

Returns context dict

get_partitial_list(**args*, ***kwargs*)

Returns a dict with keys as selectors and values as partitial file paths :return: dict :Example: {"#foo": "myapp/partitials/foo.html"}

partitial_list = {}

Define here all partitial html files used in the template_name File Write as follows: {"#foobar": "myapp/partitial/foo.html"}

class partitialajax.mixin.**UpdatePartitialAjaxMixin**

class partitialajax.mixin.**PartitialAjaxMixin**
Add this Mixin to your View to activate the useage of PartitialAjax

CHAPTER 9

PartitialAjax

class PartitialAjax(*options*, *event*)

Arguments

- **options** – dict with configuration options for each seperate Partitial. See: *Options*
- **event** – dict with function bindings for hooks. See *Events*

getCookie(*name*)

Get Information from given cookie (used for csrf)

Arguments

- **name** – cookieName

Returns **null** –

jsconsole(*level*, *info*)

Unsued console infom method

Arguments

- **level** – Level: (all console.* levels)
- **info** – Text message

camelToKebab(*string*)

Converts camelCase to kebab-case

Arguments

- **string** – camelCase String

Returns **string** – converted kebab-case string

kebabToCamel(*string*)

Converts kebab-case to camelCase string

Arguments

- **string** – kebab-case-string

Returns **String|void|*** – new converted camelCaseString

9.1 Options

Available Options:

- url
- element
- onlyChildReplace
- interval
- allowedElements
- textEventCallback
- restrictRemoteConfiguration
- configFromElement
- directLoad

9.2 Events

Available Events:

- afterSetup (constructor)
- onRemoteError
- onRemoteData
- onHandeldRemoteData
- onResponseError

CHAPTER 10

Template Tags

partitialajax.templatetags.partitialajax.**direct_partitial**(*context*, *selector*, ***kwargs*)

Embedds a Partitial without initiial loading, or updates

Parameters

- **context** – (Automatic Argument) Render Context
- **selector** – String which partitial should be included. (Same as defined in Partitial List)
- **kwargs** – Possible kwargs: allowed_methods, reload, url

Return context new dict with rended context

CHAPTER 11

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

partitialajax.mixin, 17
partitialajax.templatetags.partitialajax, 21

Index

C

camelToKebab() (*built-in function*), 19
CreatePartitialAjaxMixin (*class in partialajax.mixin*), 17

D

DeletePartitialAjaxMixin (*class in partialajax.mixin*), 17
DetailPartitialAjaxMixin (*class in partialajax.mixin*), 17
direct_partitial() (*in module partialajax.templatetags.partialajax*), 21

G

get_partitial() (*partialajax.mixin.PartitialAjaxMixin method*), 17
get_partitial_context() (*partialajax.mixin.PartitialAjaxMixin method*), 17
get_partitial_list() (*partialajax.mixin.PartitialAjaxMixin method*), 17
getCookie() (*built-in function*), 19

J

jsconsole() (*built-in function*), 19

K

kebabToCamel() (*built-in function*), 19

L

ListPartitialAjaxMixin (*class in partialajax.mixin*), 17

P

partitial_list (*partialajax.mixin.PartitialAjaxMixin attribute*), 17
PartitialAjax() (*class*), 19
partialajax.mixin (*module*), 17
partialajax.templatetags.partialajax (*module*), 21
PartitialAjaxMixin (*class in partialajax.mixin*), 17, 18

U

UpdatePartitialAjaxMixin (*class in partialajax.mixin*), 17