django-tokenfield Documentation

Release 0.1.0.dev

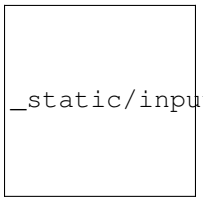
Bradley Ayers

Sep 12, 2017

Contents

django-tokenfield provides a form input that behaves like a normal text input, but allows arbitrary placeholder variables/tokens to be inserted at any position in the text. It's inspired by Facebook's message recipient input.

Example:



CHAPTER 1

Installation

1. Download and install django-tokenfield – `pip install django-tokenfield`
2. Add `django_tokenfield` to `INSTALLED_APPS` in your project's `settings.py`.
3. When rendering a form that contains a `TokenField`, ensure to include `form.media`, jQuery, jQuery templating, and `knockout.js`.

CHAPTER 2

The Widget

The widget handles translating the Python representation of a token string into HTML, JS, and CSS that the browser can render into an interactive input which is capable of sending back a suitably encoded representation of the user's input.

In Django the `Widget.render()` method must be capable of accepting data in the following formats:

- JSON string:

```
u'[
    {"type": "literal", "value": "abc"},
    {"type": "variable", "value": "firstName"},
    ...
]'
```

todo

When does this occur?

- Python token string representation:

```
[<LiteralToken object>, <VariableToken object>, ...]
```

- Junk string:

```
u'{]l#INAWD}'
```

A junk string may occur when a field is considered invalid, in which case the raw POST data is passed back to `Widget.render()` unmodified. The only situation when this would occur is if the JavaScript front-end is broken. The behaviour in these situations is to revert to an *empty* value.