

MkDocs

MkDocs es un generador de sitios estáticos rápido , simple y francamente magnífico que está orientado a la creación de documentación de proyectos. Los archivos de origen de la documentación se escriben en Markdown y se configuran con un solo archivo de configuración YAML. Comience leyendo el tutorial introductorio , luego consulte la Guía del usuario para obtener más información.

Características

Grandes temas disponibles

- Hay una pila de temas atractivos disponibles para MkDocs. Elija entre los temas incorporados: mkdocs y readthedocs , seleccione uno de los temas de terceros que se enumeran en la página wiki de Temas de MkDocs o cree el suyo propio .

Fácil de personalizar

- Consiga que la documentación de su proyecto tenga el aspecto que desea personalizando su tema y/o instalando algunos complementos . Modifique el comportamiento de Markdown con las extensiones de Markdown . Muchas opciones de configuración están disponibles.

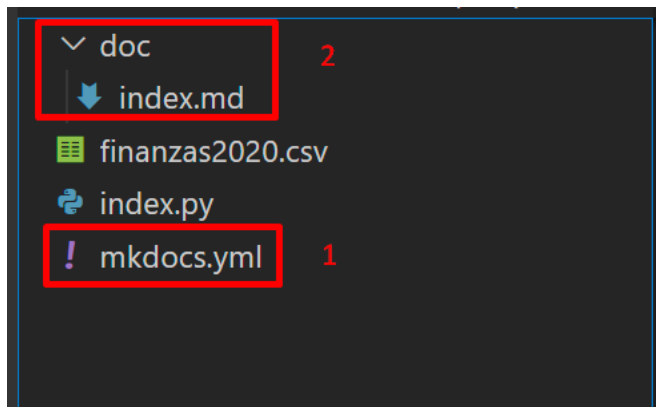
Obtenga una vista previa de su sitio mientras trabaja

- El servidor de desarrollo incorporado le permite obtener una vista previa de su documentación mientras la escribe. Incluso recargará automáticamente y actualizará su navegador cada vez que guarde sus cambios.

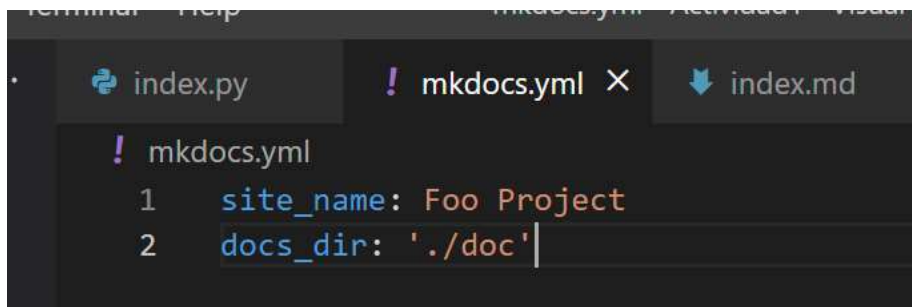
Anfitrión en cualquier lugar

- MkDocs crea sitios HTML completamente estáticos que puede alojar en páginas de GitHub, Amazon S3 o en cualquier otro lugar que elija.

Ejemplo:



1- Es el fichero de configuración de Mkdocs



2- Fichero de documentación en el cual mediante markdown, podemos ir documentando nuestro código.

index.py

! mkdocs.yml

index.md X

doc > index.md

```
172  ##Calcular gasto Total del año
173  ...
174  def gastoTotal(df):
175      total = 0
176
177      for mes in df.columns:
178          total += calcularGastosMensual(df, mes)
179
180      print('El gasto total es: ' + str(total))
181  ...
182  ##Calcular ingresos totales del año.
183  ...
184  def ingresosTotal(df):
185      total = 0
186
187      for mes in df.columns:
188          total += calcularIngresosMensual(df, mes)
189
190      print('Ingresos total es: ' + str(total))
191  ...
192
193
194  ##Método que ejecuta todo el proceso.
195  ...
196  def calcular():
197      try:
198          df = inicializarDataFrame()
199          # validamos todos los datos
200          validarTodosLosDatos(df)
201
202          calcularMesMasGastos(df)
203          calcularMesMasAhorro(df)
204          mediaGastos(df)
205          gastoTotal(df)
206          ingresosTotal(df)
207
208      except FileNotFoundError:
209          print('El fichero no se ha encontrado.')
210      except ErrorNumeroColumnas as error:
211          print(error)
```

Al lanzar el comando **python -m mkdocs serve** podremos visualizar en el navegador el resultado.

Definimos errores personalizados

```
class ErrorNumeroColumnas(Exception):
    def __init__(self, value):
        self.value = value

class ErrorMesNoContenido(Exception):
    def __init__(self, value):
        self.value = value
```

Valida el dato proporcionado como parametro

```
def validarDato(data, mes):
    if not(data.replace('\n', '').strip().isdigit()):
        raise ValueError("El valor '{}' a dato + '{\n' del mes '{}' no es válido")
    if not(data.isdigit()):
        raise ValueError("El valor '{}' a dato + '{\n' del mes '{}' no es válido, pero como se puede traducir a un número")
```

Método que valida los datos de todos los meses

```
def validarTodosLosDatos(df):
    for mes in df.columns:
        try:
            for fila in df.index:
                valorFila = df.loc[fila]
                validarDato(valorFila, mes)
        except ValueError as error:
            print(error)
            continue
```

Inicializa el dataframe obteniendo los datos del csv para sanitizarlos y generar un nuevo dataframe con dichos datos.

```
def inicializarDataFrame():
    # Copiamos el fichero sin limpiar
    dfOriginal = pd.read_csv('datosSinLimpiar.csv')

    # Definimos nuevas columnas
    headers = ''.join(dfOriginal.columns.values).split('\n')

    if len(headers) != 12:
        raise ErrorNumeroColumnas("El csv debe tener 12 columnas.")

    # Creamos un nuevo dataframe al cual almacenar los datos limpiados
    dfLimpiado = pd.DataFrame(columns=headers)

    # Recorremos cada row
    for row in dfOriginal.iterrows():

        # Obtenemos los valores para cada fila
        valorLimpiado = row[1].split("\n")
```