

# chrisvalidation

---

## 目錄

- [chrisvalidation](#)
  - [目錄](#)
  - [一、介紹及使用](#)
    - [1-1: 介紹](#)
    - [1-2: 驗證快速入門](#)
    - [1-3: 定義檔案路徑及架構](#)
    - [1-4: 第一支測試api](#)
  - [二、可用的驗證規則](#)
    - [accepted\(接受\)](#)
      - [實作方式](#)
    - [accepted\\_if\(如果...接受\):\*filed, value, ...\*](#)
      - [實作方式](#)
    - [active\\_url\(真實URL\)](#)
      - [實作方式](#)
    - [array\(陣列\)](#)
      - [實作方式](#)
    - [bail\(停止\)](#)
      - [實作方式](#)
    - [boolean|bool\(布林值\)](#)
      - [實作方式](#)
    - [in\(包含\):\*valuelist\*](#)
      - [實作方式](#)
    - [integer|int\(整數\)](#)
      - [實作方式](#)
    - [ip](#)
      - [實作方式](#)
    - [ipv4](#)
      - [實作方式](#)
    - [ipv6](#)
      - [實作方式](#)
    - [json](#)
      - [實作方式](#)
    - [max\(小於\):\*value{int}\*](#)
      - [實作方式](#)
    - [min\(大於\):\*value{int}\*](#)
      - [實作方式](#)
    - [not\\_regex\(非正規表達式\):\*value{regex}\*](#)
      - [實作方式](#)
    - [nullable\(可空\)](#)
      - [實作方式](#)
    - [regex\(正規表達式\):\*value{regex}\*](#)

- 實作方式
  - required(必填的)
    - 實作方式
  - size(大小):*value{int}*
    - 實作方式
  - string|str(字串)
    - 實作方式
- 十二、註解及參見
  - 註解
  - 參見

# 一、介紹及使用

## 1-1: 介紹

chrisvalidation提供了多種不同的方法來驗證應用程式的傳入資料。最常見的是使用validate所有傳入 HTTP 請求上可用的方法。但是，我們也將討論其他驗證方法。

chrisvalidation包含各種可應用於資料的便利驗證規則，甚至提供驗證給定資料庫表中值是否唯一的能力。我們將詳細介紹每個驗證規則，以便您熟悉 chrisvalidation 的所有驗證功能。

## 1-2: 驗證快速入門

為了快速了解 chrisvalidation 強大的驗證功能，讓我們來看看一個驗證表單並向使用者顯示錯誤訊息的完整範例。透過閱讀這個進階概述，您將能夠很好地了解如何使用 chrisvalidation 驗證傳入的請求資料：

## 1-3: 定義檔案路徑及架構

首先，假設我們的檔案包含以下架構(使用django當範例)：

```
--backend/  
|--- backend/  
|   |--- urls.py  
|   |--- setting.py  
|   |--- ....  
|--- api  
|   |--- api.py  
|   |--- urls.py  
|   |--- ....  
|--- function  
|   |--- validate.py (此函數擺放位置!! 當然你可以擺在其他地方，只要注意路徑是對的就好)
```

django內容此處將略過僅說明有關本驗證器的部分。

在function/validate.py中將包含在[readme](#)中下載的程式碼。

## 1-4: 第一支測試api

在api/api.py中(以登入api為例 然後帳號:admin 密碼:1234)

```
import json  
from rest_framework import status  
from rest_framework.decorators import api_view  
from rest_framework.response import Response  
  
from function.validation import * # 函式引入在這!  
  
@api_view(["POST"])  
def signin(request):  
    data=validate(json.loads(request.body),{
```

```

        "username": "required|string",
        "password": "required|string"
    },{
        "required": "ERROR_requestdata_not_found",
        "string": "ERROR_requestdata_type_error"
    })

if data["success"]:
    username=data["data"]["username"]
    password=data["data"]["password"]
    if username=="admin":
        if password=="1234":
            return Response({
                "success": True,
                "data": {
                    "token": "user_token",
                    "userid": "1",
                    "permission": "admin",
                    "name": "chris"
                }
            },status.HTTP_200_OK)
        else:
            return Response({
                "success": False,
                "data": "ERROR_password_error"
            },status.HTTP_401_UNAUTHORIZED)
    else:
        return Response({
            "success": False,
            "data": "ERROR_username_error"
        },status.HTTP_401_UNAUTHORIZED)
else:
    return Response({
        "success": False,
        "data": data["error"]
    },status.HTTP_400_BAD_REQUEST)

```

## 二、可用的驗證規則

以下是所有可用驗證規則及其功能的清單：

accepted(接受) accepted\_if(如果...接受) active\_url(真實URL) array(陣列) bail(停止) boolean(布林值) max(小於) in(包含) interger(整數) ip ipv4 ipv6 JSON min(大於) not\_regex(正規表達式) nullable(可空) regex(正規表達式) required(必需的) size(大小) string(字串)

### accepted(接受)

驗證的欄位必須是"yes"、"on"、1、"1"、true或"true"。這對於驗證「服務條款」接受或類似欄位很有用。

### 實作方式

依照給定規則判斷。

程式碼:

```
if value not in ["yes", "on", 1, "1", True, "true"]:  
    return seterror(testkey, rulename)
```

---

**accepted\_if(如果...接受):***filed, value, ...*

驗證中的欄位必須是"yes"、"on"、1、"1"、true或"true"如果驗證中的另一個欄位等於指定值。這對於驗證「服務條款」接受或類似欄位很有用。

**實作方式**

依照給定規則判斷。

程式碼:

```
if len(rulevaluelist)!=2:  
    return seterror(testkey, rulename)  
otherkey=rulevaluelist[0]  
othervalue=rulevaluelist[1]  
othertarget=getvaluebypath(data, otherkey)  
if othertarget==othervalue:  
    if value not in ["yes", "on", 1, "1", True, "true"]:  
        return seterror(testkey, rulename)
```

---

**active\_url(真實URL)**

驗證下的欄位必須根據socket.gethostbyname函數具有有效的 A 或 AAAA 記錄。

**實作方式**

依照給定規則判斷。

程式碼:

```
try:  
    host=re.sub(r"^https?:://", "", value).split("/")[0]  
    socket.gethostbyname(host)  
except:  
    return seterror(testkey, rulename)
```

---

**array(陣列)**

驗證下的欄位必須是array。也就是需符合List型別。

## 實作方式

依照給定規則判斷。

程式碼:

```
if not isinstance(value,list):
    return seterror(testkey,rulename)
```

---

## bail(停止)

第一次驗證失敗後停止執行該欄位的驗證規則。

當bail規則僅在遇到驗證失敗時才會停止驗證特定字段，您可以使用函數內的第四個參數**checkall=True**，一旦發生單一驗證失敗，它應該停止驗證所有屬性。

範例:

```
validate(data={
    "key": 123
},rule={
    "key": "bail|required|string|min:2"
},error={
    "bail": "ERROR_bail",
    "required": "ERROR_required",
    "string": "ERROR_type_string",
    "min": "ERROR_min_length"
},checkall=True)
```

## 實作方式

依照給定規則判斷。

程式碼:

```
bailstop=False
for testrule in testrulelist:
    if bailstop:
        break

    returndata=test(fullkey,testrule,value)

    if not returndata["check"]:
        check=False
        errordata[fullkey]={}
```

```

        errordata[fullkey]
[returndata["rulename"]]=returndata["errordata"].replace(":key",f" '{fullkey.split(
".")[-1]}'")
        if not firsterror:

firsterror=returndata["errordata"].replace(":key",f" '{fullkey.split(".")[-1]}'")
        if not checkall:
            break
        if "bail" in testrulelist:
            bailstop=True

```

## boolean|bool(布林值)

驗證下的欄位必須能夠轉換為布林值。接受的輸入是true、false、1、0、"1"、"0"

### 實作方式

依照給定規則判斷。

程式碼:

```

if not isinstance(value,bool) and value not in [0,1,"0","1"]:
    return seterror(testkey,rulename)

```

## in(包含):valuelist

驗證字串是否包含在給定的值清單中(以逗號分隔)。

當代驗證物為array時，輸入陣列中的每個值都必須存在於給定清單中。

### 實作方式

依照給定規則判斷。

程式碼:

```

allowed=rulevalue.split(",")
if isinstance(value,list):
    for key in value:
        if str(key) not in allowed:
            return seterror(testkey,rulename)
else:
    if str(value) not in allowed:
        return seterror(testkey,rulename)

```

in\_array:另一個字段.\* 驗證下的欄位必須存在於anotherfield的值中。

---

## integer|int(整數)

驗證的欄位必須是整數。

此驗證規則不驗證輸入是否屬於「整數」變數類型，僅驗證輸入是否屬於 `PHPFILTER_VALIDATE_INT`規則接受的類型。如果您需要驗證輸入是否為數字，請將此規則與驗證規則結合 `numeric` 使用。

### 實作方式

驗證是否為整數。

程式碼:

```
if not isinstance(value,int) and not isinstance(value,float):  
    return seterror(testkey,rulename)
```

---

## ip

驗證的欄位必須是IP地址。

### 實作方式

依給定要求驗證。

程式碼:

```
try:  
    ipaddress.ip_address(value)  
except:  
    return seterror(testkey,rulename)
```

## ipv4

驗證下的欄位必須是IPv4地址。

### 實作方式

依給定要求驗證。

程式碼:

```
try:  
    if not isinstance(ipaddress.ip_address(value), ipaddress.IPv4Address):  
        return seterror(testkey,rulename)
```



```
except:
    return seterror(testkey,rulename)
```

---

## ipv6

驗證下的欄位必須是IPv6地址。

### 實作方式

依給定要求驗證。

程式碼:

```
try:
    if not isinstance(ipaddress.ip_address(value), ipaddress.IPv6Address):
        return seterror(testkey,rulename)
except:
    return seterror(testkey,rulename)
```

## json

驗證下的欄位必須是json。也就是需符合Dictionary型別。

### 實作方式

依照給定規則判斷。

程式碼:

```
if not isinstance(value,dict):
    return seterror(testkey,rulename)
```

---

## max(小於):value{int}

驗證的欄位必須小於或等於給定值。字串、數字、陣列和檔案的評估方式與size規則相同。

### 實作方式

先用checksize函數判斷輸入長度(大小)，然後使用給定值驗證是否小於輸入長度(大小)。

程式碼:

```
size=checksize(value)
try:
```

```
if size==False or int(rulevalue)<size:
    return seterror(testkey,rulename)
except:
    return seterror(testkey,rulename)
```

---

## min(大於):value{int}

驗證的欄位必須大於給定值。字串、數字、陣列和檔案的評估方式與size規則相同。

### 實作方式

先用checksize函數判斷輸入長度(大小)，然後使用給定值驗證是否大於輸入長度(大小)。

程式碼:

```
size=checksize(value)
try:
    if size==False or size<int(rulevalue):
        return seterror(testkey,rulename)
except:
    return seterror(testkey,rulename)
```

---

## not\_regex(非正規表達式):value{regex}

驗證的欄位不能與給定的正規表示式相符。

在內部，此規則使用 PHPpreg\_match函數。指定的模式應遵循所需的相同格式preg\_match，因此也包括有效的分隔符號。例如：'email' => 'not\_regex:/^.+\$/i'。

當使用regex/not\_regex模式時，可能需要使用陣列而不是使用|分隔符號來指定驗證規則，特別是當正規表示式包含|字元時。

### 實作方式

因python無法直接將rulevalue丟入，所以會先驗證是哪種類型 // -> search /^\/\.\$/ -> match /^\$/ -> fullmatch 然後在做驗證

程式碼:

```
if type(value)!=str:
    return seterror(testkey,rulename)

pattern=rulevalue

if pattern.startswith("/") and pattern.rfind("/")>0:
    lastslash=pattern.rfind("/")
    regexbody=pattern[1:lastslash]
```

```
    flags=pattern[lastslash+1:]
    flagval=0
    if "i" in flags:
        flagval|=re.IGNORECASE
    if "m" in flags:
        flagval|=re.MULTILINE
    if "s" in flags:
        flagval|=re.DOTALL
    pattern=regexbody
else:
    flagval=0

try:
    regex=re.compile(pattern,flagval)
except:
    return seterror(testkey,rulename)

if regex.search(value):
    return seterror(testkey,rulename)
```

---

## nullable(可空)

正在驗證的欄位可能是null。

### 實作方式

若請求值為null，則不進行其他驗證。

程式碼:

```
if not (("nullable" in testrulelist) and (value==None)):
    # 正常的驗證規則在這
    # ...
```

---

## regex(正規表達式):value{regex}

驗證下的欄位必須與給定的正規表示式相符。

在內部，此規則使用regex函數。指定的模式應遵循所需的相同格式match，因此也包括有效的分隔符號。例如："email" => "regex:/^.+@.+\$ /i"。

當使用regex/not\_regex模式時，可能需要在陣列中指定規則而不是使用|分隔符，特別是當正規表示式包含|字元時。

### 實作方式

因python無法直接將rulevalue丟入，所以會先驗證是哪種類型 // -> search /^\/\$/ -> match /^\$/ -> fullmatch 然後在做驗證

程式碼:

```
if type(value)!=str:
    return seterror(testkey,rulename)

pattern=rulevalue

if pattern.startswith("/") and pattern.rfind("/")>0:
    lastslash=pattern.rfind("/")
    regexbody=pattern[1:lastslash]
    flags=pattern[lastslash+1:]
    flagval=0
    if "i" in flags:
        flagval|=re.IGNORECASE
    if "m" in flags:
        flagval|=re.MULTILINE
    if "s" in flags:
        flagval|=re.DOTALL
    pattern=regexbody
else:
    flagval=0

try:
    regex=re.compile(pattern,flagval)
except:
    return seterror(testkey,rulename)

if not regex.search(value):
    return seterror(testkey,rulename)
```

---

## required(必填的)

驗證欄位必須存在於輸入資料中且不能為空。如果欄位符合以下條件之一，則該欄位為「空」：

- 值為null。
- 該值為空字串。
- 該值是一個空數組或空物件。
- 該值是一個沒有路徑的上傳檔案。

## 實作方式

依照給定規則判斷。

程式碼:

```
if value is None or value==" or value==[] or value=={}:  
    return seterror(testkey,rulename)
```

---

### size(大小):value{int}

驗證欄位的大小必須與給定的值相符。對於字串數據，值對應於字元數。對於數值數據，值對應於給定的整數值（屬性也必須具有numeric或integer規則）。對於數組來說，大小對應於count數組的。對於文件，大小對應於文件大小（以千位元組為單位）。讓我們來看一些例子：

```
# 驗證字串長度為12  
"title": "size:12";  
  
# 驗證數值為10  
"seats": "integer|size:10";  
  
# 驗證陣列長度為5個元素  
"tags": "array|size:5";  
  
# 驗證文件大小為512KB  
"image": "file|size:512";
```

### 實作方式

依照給定規則判斷。

程式碼:

```
def checksize(value):  
    if isinstance(value,str):  
        return len(value)  
    elif isinstance(value,int) or isinstance(value,float):  
        return value  
    elif isinstance(value,list):  
        return len(value)  
    elif isinstance(value,dict) and "size" in value:  
        return value["size"]  
    else:  
        return False  
  
size=checksize(value)  
try:  
    if size==False or size<int(rulevalue):  
        return seterror(testkey,rulename)  
except:  
    return seterror(testkey,rulename)
```

## string|str(字串)

驗證的字段必須是字串。如果您希望允許該欄位也允許null，則應將nullable規則指派給該欄位。

### 實作方式

依照給定規則判斷。

程式碼:

```
if not isinstance(value,str):  
    return seterror(testkey,rulename)
```

## 十二、註解及參見

註解

參見

20250707 v001000003