```
# Hello World project
# python3
#  3 تهيه فايل پروژه اي جديدازكلاس پايتون
# مرجع انتشاردر git hap
#
https://github.com/emmetio/py-emmet/blob/master/emm
et/css_abbreviation/tokenizer/tokens.py


class Token:
    __slots__ = ('start', 'end')

    def __init__(self, start: int=None, end:
int=None):
        self.start = start
        self.end = end

    @property
    def type(self):
        "Type of current token"
        return self.__class__.__name__

    def to_json(self):
        return dict([(k, self.__getattribute__(k))
for k in dir(self) if not k.startswith('__') and k
!= 'to_json'])

class Chars:
```

```
    Hash = '#'
    Dollar = '$'
    Dash = '-'
    Dot = '.'
    Colon = ':'
    Comma = ','
    Excl = '!'
    At = '@'
    Percent = '%'
    Underscore = '_'
    RoundBracketOpen = '('
    RoundBracketClose = ')'
    CurlyBracketOpen = '{'
    CurlyBracketClose = '}'
    Sibling = '+'
    SingleQuote = "'"
    DoubleQuote = '"'
    Transparent = 't'

class OperatorType:
    Sibling = '+'
    Important = '!'
    ArgumentDelimiter = ','
    ValueDelimiter = '-'
    PropertyDelimiter = ':'

class Operator(Token):
    __slots__ = ('operator',)

    def __init__(self, operator: OperatorType,
```

```
*args):
        super(Operator, self).__init__(*args)
        self.operator = operator


class Bracket(Token):
    __slots__ = ('open',)

    def __init__(self, is_open: bool, *args):
        super(Bracket, self).__init__(*args)
        self.open = is_open

class Literal(Token):
    __slots__ = ('value',)

    def __init__(self, value: str, *args):
        super(Literal, self).__init__(*args)
        self.value = value

class NumberValue(Token):
    __slots__ = ('value', 'raw_value', 'unit')

    def __init__(self, value: int, raw_value: str,
unit='', *args):
        super(NumberValue, self).__init__(*args)
        self.value = value
        self.raw_value = raw_value
        self.unit = unit

class ColorValue(Token):
```

```
    __slots__ = ('r', 'g', 'b', 'a', 'raw')

    def __init__(self, r=0, g=0, b=0, a=None,
raw='', *args):
        super(ColorValue, self).__init__(*args)
        self.r = r
        self.g = g
        self.b = b
        self.a = a if a is not None else 1
        self.raw = raw

class StringValue(Token):
    __slots__ = ('value', 'quote')

    def __init__(self, value: str, quote='',
*args):
        super(StringValue, self).__init__(*args)
        self.value = value
        self.quote = quote

class Field(Token):
    __slots__ = ('name', 'index')

    def __init__(self, name: str, index: int=None,
*args):
        super(Field, self).__init__(*args)
        self.index = index
        self.name = name

class WhiteSpace(Token): pass
```