Metodika štýlu písania kódu pre Python

Metodika štýlu písania kódu pre Python

Tím č. 03 FireAnt

Vedúci tímu: Branislav Pecher, Ivan Srba

Autor metodiky: Denis Mitana

Dátum vytvorenia: 11.10.2019

Poslednú zmenu vykonal: Denis Mitana

Dátum poslednej zmeny: 19.10.2019

Obsah

Úvod

Pre zachovanie konzistencie v projekte budeme nasledovať konvencie určené štandardom PEP 8 pre štýlovanie kódu a štandardom PEP 257 pr e konvencie dokumentačných reťazcov. Keďže tieto štandardy sú rozsiahle, tak v nasledujúcich sekciách opíšeme najdôležitejšie pravidlá.

Odporučené vývojové prostredie (IDE) je PyCharm, ktoré poskytuje aj profesionálnu verziu v študentskej licencii. PyCharm má od výroby zapnutú inšpekciu kódu podľa PEP 8 a viaceré nástroje, ktoré pomáhajú automaticky formátovať kód, aby bol podľa standardov.

POZOR: Primárne sa riadime podľa pravidiel uvedených v tomto dokumente. PEP 8 a PEP 257 sú odporúčania, ktoré niekedy opisujú viaceré prípustné spôsoby a preto pravidlá v tomto dukumente sú nadradené odporúčaniam PEP 8 a PEP 257.

Usporiadanie kódu

Pravidlá usporiadania kódu.

Odsadenie

Úroveň odsadenia je definovaná **4 medzerami**.

Nasledujúca ukážka definuje štýl odsádzania ak sa (key word) argumenty nezmestia do jedného riadku, resp. ak by sa aj zmestili, ale chceme ich zámerne odsadiť.

Štýly odsadzovania

Štýly odsadzovania

```
d4 = {
    'key1': 'value',
    'key2': 'value',
    'key3': 'value',
    'key4': 'value',
    'key5': 'value'
}
```

Tabulátory vs. medzery

Na odsádzanie **vždy** používame medzery.

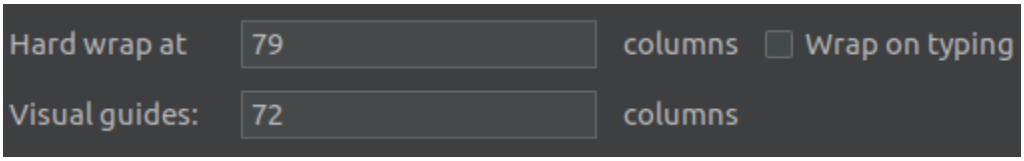
Odporúčane je si v PyCharm-e nastaviť TAB na 4 medzery.

Maximálna dĺžka riadku

Maximálna povolená dĺžka riadku je **79** znakov pre všetky riadky. Pre dlhé textové bloky (dokumentačné reťazce alebo komentáre) je povolené maximálne **72** znakov.

Nastavenie maximálnej dĺžky riadku v PyCharm-e

Settings -> Code Style



Prázdne riadky

Funkcie a definície tried sa oddeľujú **2** prázdnymi riadkami.

Definície metód v triedach sa oddeľujú **1** prázdnym riadkom.

Na konci súboru musí byť **1** prázdny riadok.

Reťazce

Úvodzovky pri reťazcoch

Reťazce sa uvádzajú do **apostrofov (')**. Reťazec, ktorý obsahuje apostrof budeme uvádzať do **úvodzoviek (")**. Dokumentačné reťazce uvádzame vždy do **úvodzoviek (")**.

Formátovanie reťazcov

Na formátovanie reťazcov budeme používať momentálne najnovšiu a nejlepšiu techniku navývanú **f-strings**.
Príklad formátovania reťazcov pomocou f-string

Príklad formátovania reťazcov pomocou f-string

```
name = 'Eric'
age = 74
print(f'Hello, {name}. You are {age}.')
```

Medzery medzi členmi a operátormi

Medzi každým operátorom a operandom **bude** medzera.

Príklad medzier v matematických výrazoch

```
# No
x_1 = (- b+(b**2-4*a*c)**(1/2)) / (2*a)

# Yes
x_1 = (- b + (b**2 - 4 * a * c)**(1 / 2)) / (2 * a)
```

Medzery uvádzame tiež medzi argumentami funkcie a medzi jednotlivými prvkami zoznamov (list, tuple, dictionary).

Pozor, pri argumentoch s predvolenými hodnotami nedávame medzi znak '=' medzery!

Príklad medzier pri zoznamoch a argumentoch funkcií

```
# No
def example(arg1,arg2,arg3=10):
    # Code here

example_list = [1,2,"example"]


# Yes
def example(arg1, arg2, arg3=10):
    # Code here

example_list = [1, 2, "example"]
```

Komentáre

Komentáre v kóde

Komentáre píšeme v **anglickom jazyku**.

Komentáre by mali byť kompletné vety, ktoré začínajú s **veľkým písmenom**. Ak veta začína identifikátorom, tak sa nemusí nutne začínať veľkým písmenom, platí, že nemeníme názvy identifikátorov. Pri jednej vete nemusíme ukončiť vetu bodkou.

Treba dvakrát zvážiť použitie komentára v riadku (prednosť majú blokové komentáre). Ak sa rozhodneme použiť komentár v riadku, tak musí byť oddelený **dvomi** medzerami a taktiež začína s **velkým písmenom**.

Dokumentačné reťazce

Každá trieda, metóda a funkcia bude opísaná pomocou dokumentačného reťazca (docstring). V docstringu pre triedu budu opísané aj jej **verejné atribúty**.

Docstring uvádzame medzi **3 úvodzovky (")**.

Pri opisovaní parametrov/atribútov/návratovej hodnoty sa uvádza ich **typ**, **predvolená hodnota (ak existuje)**. Pri parametroch sa uvádza aj to, či je atribút **statický**. Ak je opis parametru/atribútu/návratovej hodnoty na viac riadkov, tak na všetkych riadkoch okrem prvého je odsadený 4mi medzerami.

Typy atribútov: str, int, float, list, dict, tuple, ...

Docstringy musia zodpovedať nasledujúcim príkladom.

Všeobecný vzor docstringu

```
def foo(arg1, arg2=default_value):
    """
    General description sentence.

    More specific description.

    :param arg1: type, description. This is an example
        of a long description.
    :param arg2: type (default: default_value), description.
    :return: type, description.
    """
    # Code starts here
    ...
```

Konkrétny vzor docstringu

```
def quadratic(a, b, c):
    """
    Solve quadratic equation via the quadratic formula.

    A quadratic equation has the following form:
    ax**2 + bx + c = 0

    There always two solutions to a quadratic equation: x_1 & x_2.

    :param a: float, quadratic parameter. a != 0.
    :param b: float, linear parameter.
    :param c: float, absolute parameter.
    :return:
        x_1: float, first solution.
        x_2: float, second solution.
    """
    x_1 = (- b + (b**2 - 4 * a * c)**(1 / 2)) / (2 * a)
    x_2 = (- b - (b**2 - 4 * a * c)**(1 / 2)) / (2 * a)

    return x_1, x_2
```

Pri komentovaní tried sa komentujú aj atribúty na úrovni triedy a aj atribúty na úrovni objektu. Ku atribútom na úrovni triedy sa pridáva znalosť o ich **statickosti**.

V tejto ukážke je tiež dôležité si všimnúť **jednoriadkový** docstring.

Konkrétny vzor docstringu pre triedy

```
class CentralStorageClient:
    """
    CentralStorageClient is client to perform requests on central
    storage API, handling also authorization and re-authorization.

    :param token: str (static), token to be used in authorization.
    :param token_expiration: int (static), token expiration time.
    :param username: str, username for authorization to central storage
    :param password: str, password for authorization to central storage
    """
    token = None
    token_expiration = None

    def __init__(self):
        """Create object of the class CentralStorageClient."""
        self.username = api_properties.central_storage_api_username
        self.password = api_properties.central_storage_api_password
```

Konvencie pomenovaní

Názvy balíkov a modulov

Krátke názvy, malými písmenami, neodporúča sa použitie podčiarkovníkov (_).

Názvy tried

Používame **CapitalizedWords** (CapWords, CamelCase).

Pri použití skratiek sa budú všetky písmena skratky veľkými písmenami (použijeme HTTPServerError namiesto HttpServerError).

Názvy funkcií, premenných a argumentov

Výstižne názvy, malými písmenami, na oddelenie sa použije podčiarkovník (_).

Názvy metód a inštančných premenných

Výstižne názvy, malými písmenami, na oddelenie sa použije podčiarkovník (_).

Pri neverejných metódach a atribútoch pridáme pred názov podčiarkovník (napr. self._non_public_attribute)

Názvy konštánt

Definované na úrovni modulu, všetko veľkými písmenami, na oddelenie slov sa použije podčiarkovník (napr. MAX_OVERFLOW, TOTAL, ...)