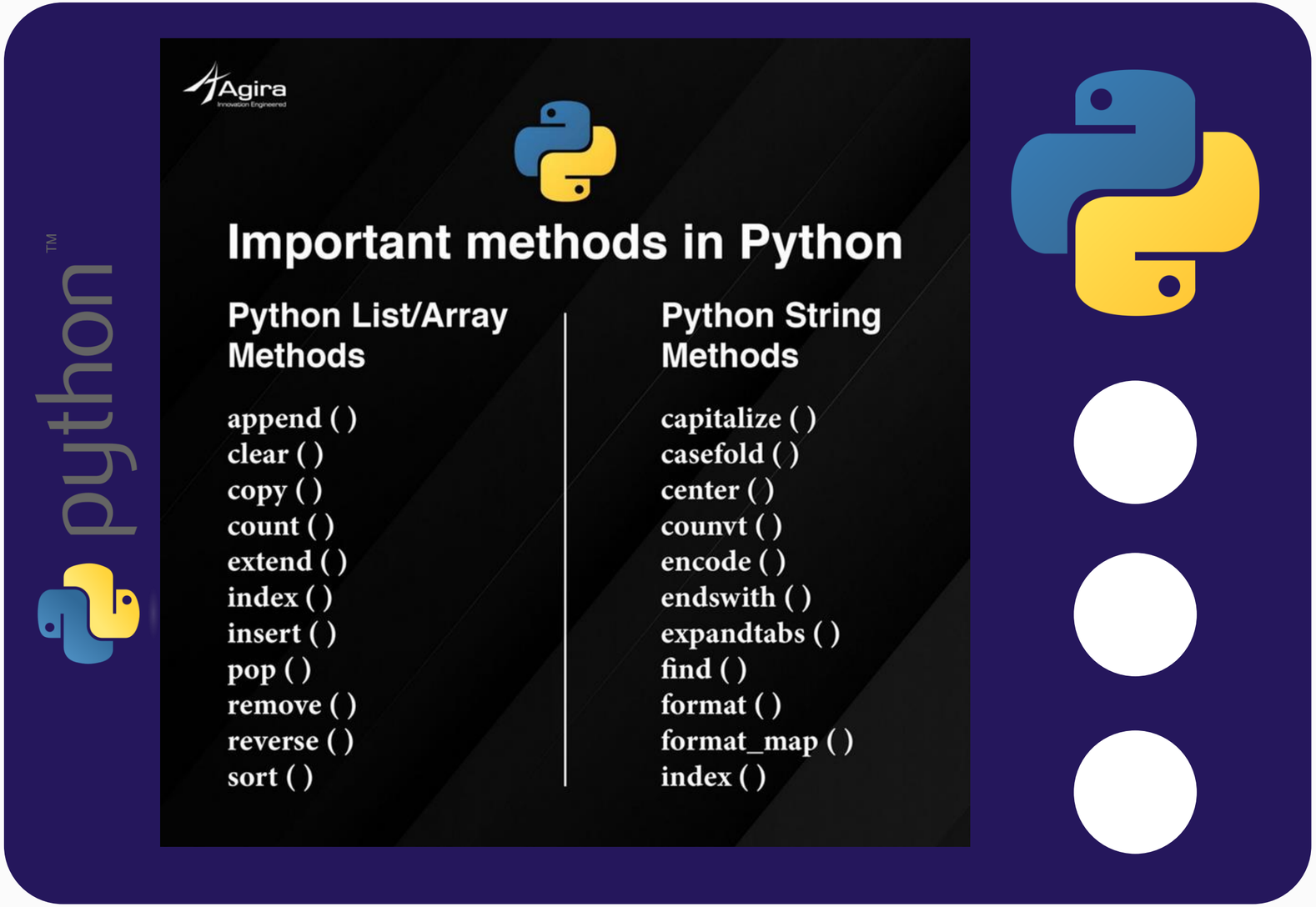
# String Method, Formatting and JSON

Penyusun Modul: Fitria Yunita Dewi
Editor: Silfa Rahma Aulia

# Method

- Method pada dasarnya adalah fungsi yang dimiliki oleh *object* tertentu (lebih jauh tentang object akan dibahas pada bab berikutnya)
- Beberapa *built-in* method yang sering digunakan:
  - list: `append()`, `insert()`, `extend()`, `sort()`, `remove()`
  - string: `lower()`, `upper()`, `strip()`, `format()`, `find()`, `capitalize()`, `isdigit()`

https://docs.python.org/3/library/stdtypes.html#mutable-sequence-types
https://docs.python.org/3/library/stdtypes.html#string-methods


Agira
Innovation Engineered
Important methods in Python
Python List/Array Methods
append ( )
clear ( )
copy ( )
count ( )
extend ( )
index ( )
insert ( )
pop ( )
remove ( )
reverse ( )
sort ( )
Python String Methods
capitalize ( )
casefold ( )
center ( )
counvt ( )
encode ( )
endswith ( )
expandtabs ( )
find ( )
format ( )
format_map ( )
index ( )
python

# String Formatting

- Salah satu metode string adalah `format()` yang merupakan salah satu dari beberapa cara string formatting
- Formatting dilakukan untuk mendekorasi string dengan berbagai variabel lainnya
- Pada Python terdapat 4 cara formatting:
    - **Format (%) operator** (*old style*)
    - **Str.format()** (*new style*)
    - **f-strings** (*strings interpolation*)
    - **Template strings** (perlu mengimport *module* `template` dan tidak akan dibahas di sini)

# Format Operator

```
placeholders:
- %s -> string
- %d -> integer
- %f -> float
    - %.<n>f -> n decimal places
```

Contoh

```
nama = 'Depok'
luas = 200.29
teks  = 'Kota %s memiliki luas sebesar %.1f km2' %(nama,luas)
print(teks)
```

✓ 0.1s

```
Kota Depok memiliki luas sebesar 200.3 km2
```

# Str.format()


## Contoh

```
nama = 'Depok'
luas = 200.29
teks  = 'Kota {0} memiliki luas sebesar {1:.1f} km2'.format(nama, luas)
print(teks)
```

✓ 0.6s

```
Kota Depok memiliki luas sebesar 200.3 km2
```

- Nilai-nilai yang akan dimasukkan ke dalam teks disiapkan di dalam `format()`
- Indeks pertama merupakan variable `nama` bertipe string, indeks kedua berisi variable `luas` bertipe *float* dengan format 1 angka desimal

# Str.format()


- Jika nilai atau variable yang menjadi argumen dalam `format()` merupakan *sequence*, datanya tetap dapat diakses menggunakan *indexing*

```
koordinat = (3, 4)
teks = "Posisi mula-mula berada pada jarak {0[0]} cm sepanjang sumbu x dan {0[1]} cm sepanjang sumbu y".format(koordinat)
print(teks)
```

```
Posisi mula-mula berada pada jarak 3 cm sepanjang sumbu x dan 4 cm sepanjang sumbu y
```

# F-string

```
nama = 'Depok'
luas = 200.29
teks = f"Kota {nama} memiliki luas sebesar {luas:.1f} km2"
print(teks)
```

✓ 0.4s

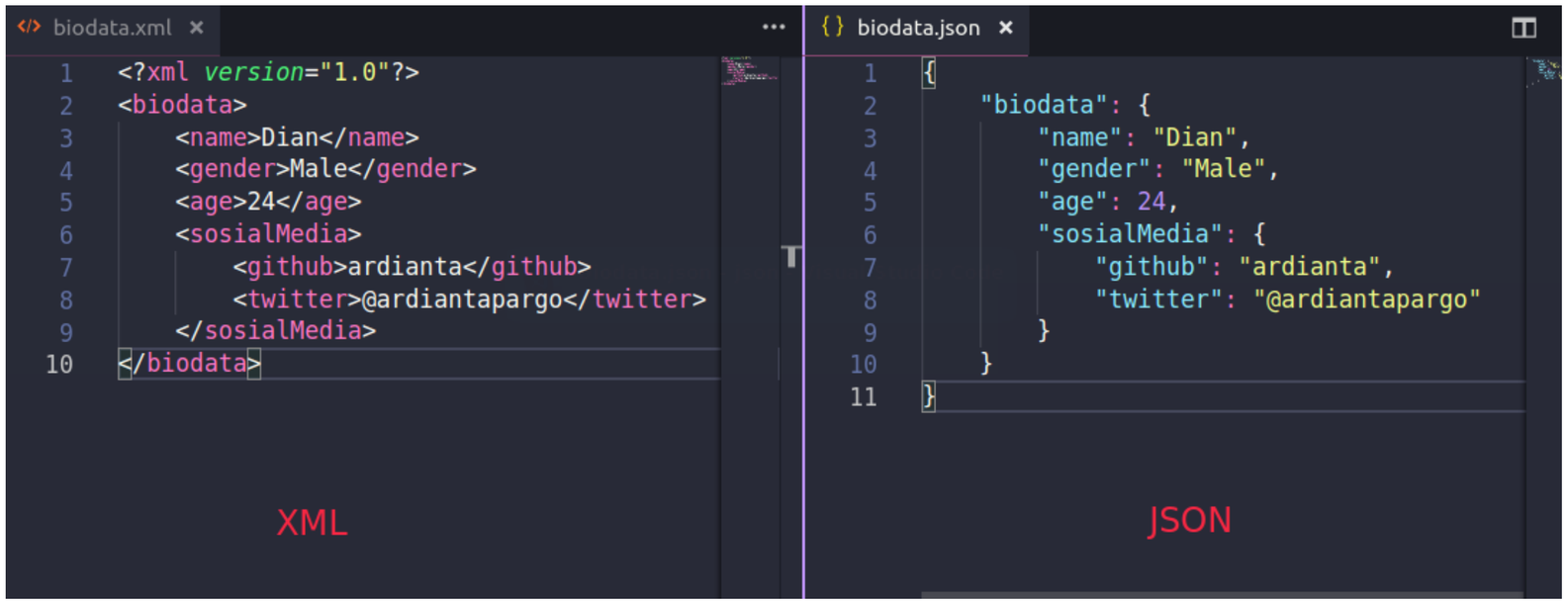
```
Kota Depok memiliki luas sebesar 200.3 km2
```

# JSON

- JSON merupakan singkatan dari JavaScript Object Notation
- Meskipun JSON adalah subset/bagian dari JavaScript, hampir seluruh bahasa pemrograman (C, C++, C#, Java, Perl, Python, dll) bisa membaca format JSON
- Apapun bahasa pemrograman yang sedang dipelajari, JSON wajib dipahami
- JSON adalah sebuah **format data** yang digunakan untuk **pertukaran dan penyimpanan data** antar aplikasi maupun antar platform pemrograman

# JSON

- Dibanding pendahulunya (XML/ eXtensible Markup Language) JSON terlihat lebih sederhana dan mudah dibaca




https://www.petanikode.com/json-pemula/

# Struktur Dasar JSON

- JSON selalu dibuka dan ditutup dengan kurung kurawal
- Terdiri dari pasangan *key* dan *value* dan pemisah tanda koma jika terdapat data lebih dari 1


Tidak sulit mempelajari JSON karena strukturnya mirip dengan tipe data *dictionary* pada Python

Namun JSON sendiri pada hakikatnya merupakan *string*

# JSON pada Python

Python memiliki *built-in package* yang dapat digunakan untuk bekerja dengan data JSON

```
import json
```

## JSON to Python: Parse JSON

*String* JSON dapat diuraikan menggunakan metode `json.loads()`

Hasilnya berupa tipe data *dictionary*

```
import json

# suatu string JSON
data_JSON = '{"nama": "Anang", "umur": 30, "kota": "Depok"}'

# parsing x
data_baru = json.loads(data_JSON)
print(data_baru)
type(data_baru)
```

```
{'nama': 'Anang', 'umur': 30, 'kota': 'Depok'}
dict
```

Jika sudah dalam tipe data dictionary maka kita dapat mengakses setiap value melalui *indexing*

```
print(data_baru["kota"])
```

```
Depok
```

# JSON pada Python

## Python to JSON

Kita dapat mengubah objek apapun pada Python menjadi string JSON menggunakan metode `json.dumps()`

```
import json

# suatu dictionary
data_dict = {"nama": "Anang", "umur": 30, "kota": "Depok"}

# ubah menjadi json
data_JSON = json.dumps(data_dict)
print(data_JSON)
type(data_JSON)
```

```
{"nama": "Anang", "umur": 30, "kota": "Depok"}

str
```

Dalam contoh di atas, dari data *dictionary* diubah menjadi *string* JSON, sehingga operasi-operasi yang bisa dilakukan pada `data_JSON` adalah operasi untuk tipe data *string*, dan operasi *dictionary* tidak lagi dapat dilakukan

# JSON pada Python

## Python to JSON

`json.dumps()` dapat juga digunakan untuk tipe data selain dictionary

```
import json

dict_to_json = json.dumps({"nama": "Anang", "umur": 30})
list_to_json = json.dumps([1, 2, "tiga"])
tuple_to_json = json.dumps(("J.A.R.V.I.S", "WALL-E"))
int_to_json = json.dumps(42)
float_to_json = json.dumps(31.76)
bool_to_json = json.dumps(True)
```

# Latihan JSON

Contoh potongan *string* JSON berikut tidak dapat dikonversi menjadi objek apapun pada Python. Temukan di mana kesalahannya!

```
import json
sampleJson = """
{
    "id":1,
    "objek":"Apel"
    "warna":[
        "merah",
        "hijau"
        ]
}"""

data = json.loads(sampleJson)
print(data)
```

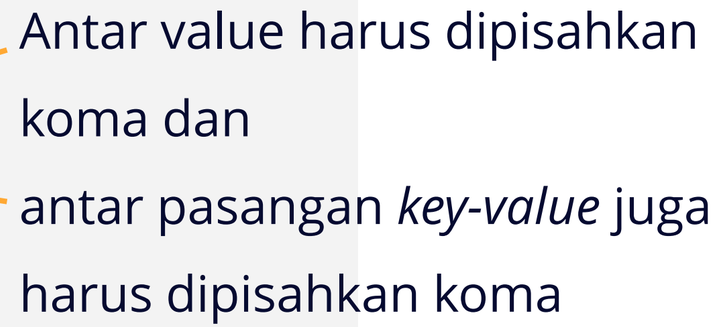
```
---------------------------------------------------------------------------
JSONDecodeError                           Traceback (most recent call last)
```

# Latihan JSON

**Solusi:**

```
import json
sampleJson = """
{
    "id":1,
    "objek":"Apel",
    "warna":[
        "merah",
        "hijau"
        ]
}"""

data = json.loads(sampleJson)
print(data)
```




```
{'id': 1, 'objek': 'Apel', 'warna': ['merah', 'hijau']}
```


thank
YOU