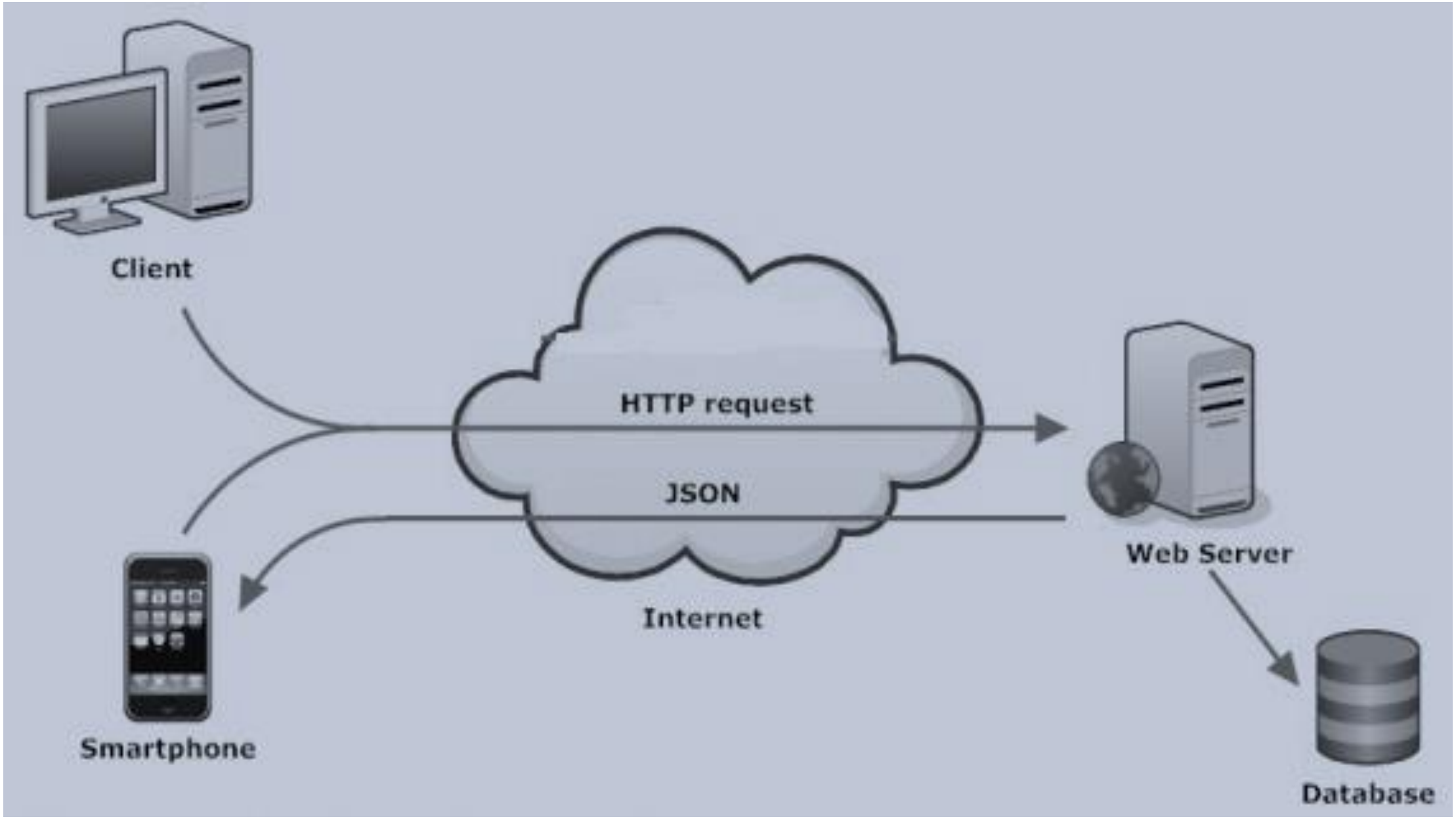
JSON

JAVA SCRIPT OBJECT NOTATION

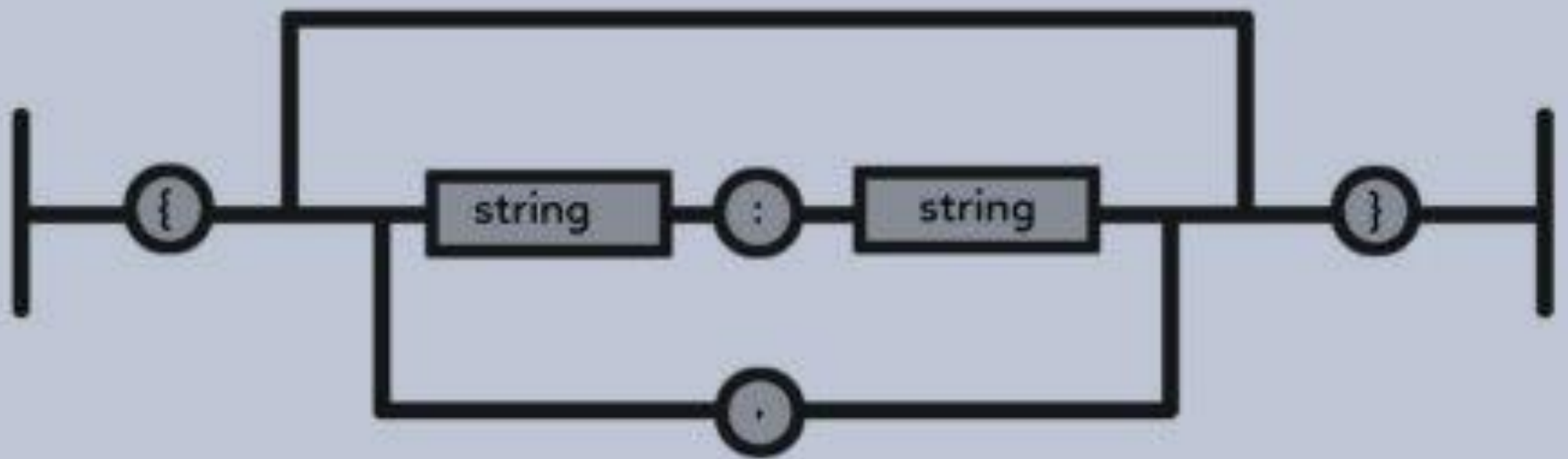

Client
HTTP request
JSON
Internet
Web Server
Smartphone
Database

- JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans and machines to read and write.

- JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python.

- JSON is a syntax for storing and exchanging data.
- The data can only be text when communicating between client and server.
- The JSON object contains an associative array of “name: value” pairs whereas the JSON array contains a sequence of values with default numeric indexes.

JSON Object Data Format



Convert from JSON to Python

```
import json
# some JSON:
j =  '{ "name":"John", "age":30, "city":"New York"}'
# parse j:
d = json.loads(j)
# the result is a Python dictionary:
print(d["age"])
```

Convert from Dictionary to JSON

```
import json
# a Python object (dict):
dct = {"name": "John","age": 30,"city": "New York“ }
# convert into JSON:
js = json.dumps(dct)
# the result is a JSON string:
print(js)
```

JSON to Dictionary

- Keys in key/value pairs of JSON are always of the type str. When a dictionary is converted into JSON, all the keys of the dictionary are coerced to strings.
- As a result of this, if a dictionary is converted into JSON and then back into a dictionary, the dictionary may not equal the original one as all fields in the original dictionary may not be strings.

Python objects to JSON strings

```
print(json.dumps({"name": "John", "age": 30}))

print(json.dumps(["apple", "bananas"]))

print(json.dumps("hello"))

print(json.dumps(42))

print(json.dumps(31.76))

print(json.dumps(True))
```

```
{"name": "John", "age": 30}

["apple", "bananas"]

"hello“

42

31.76

true
```

User Defined Dump

```
class Element :
        def __init__(self,ab,na) :
                self.name = na
                self.abbrev = ab
e = Element("H","Hydrogen")
je = json.dumps(e)
print(je)
```

builtins.TypeError: Object of type Element is not JSON serializable

__dict__

```
def jsonDefault(object):
        return object.__dict__
class Element :
        def __init__(self,ab,na) :
                self.name = na
                self.abbrev = ab
e = Element("H","Hydrogen")
je = json.dumps(e, default=jsonDefault)
print(je)
```

Prints: {"name": "Hydrogen", "abbrev": "H"}