Python Dictionaries

Quick Reference Sheet

Unit 3 - Lesson 2 - Part 3

Basic Syntax

Creating a Dictionary

```
# Empty dictionary
user = {}

# Dictionary with data
user = {
    "username": "alex",
    "age": 16,
    "verified": True
}
```

Key Rules:

- Keys must be in quotes (strings)
- Use colons `:` between keys and values
- Separate pairs with commas
- No comma after the last item (but it's okay if you do)

Accessing Values

Method	Syntax	Notes
Bracket notation	`user["username"]`	Crashes if key doesn't exist ❌
.get() method	`user.get("age")`	Returns None if missing ✅
.get() with default	`user.get("age", 0)`	Returns default if missing ✅ ✅

```
# Example
name = user["username"]     # "alex"
age = user.get("age", 0)    # 16 (or 0 if missing)
```

Best Practice:

Always use `.get()` when a key might not exist!

Modifying Dictionaries

Adding or Updating

```
# Add new key-value pair
user["email"] = "alex@example.com"

# Update existing value
user["age"] = 17
```

Removing Items

```
# Delete a key-value pair
del user["verified"]

# Remove and return value
age = user.pop("age")

# Remove all items
user.clear()
```

Common Methods

.keys()

Get all keys as a list

```
user.keys()
```

.values()

Get all values as a list

```
user.values()
```

.items()

Get key-value pairs as tuples

```
user.items()
```

.get(key, default)

Safely get a value with fallback

```
user.get("age", 0)
```

.pop(key)

Remove and return a value

```
user.pop("age")
```

.clear()

Remove all items

```
user.clear()
```

Looping Through Dictionaries

```
# Loop through keys
for key in user.keys():
    print(key)

# Loop through values
for value in user.values():
    print(value)

# Loop through both (BEST!)
for key, value in user.items():
    print(f"{key}: {value}")
```

JavaScript vs Python

JavaScript Objects

```
// Create
let user = {
  username: "alex",
  age: 16
};

// Access
user.username
user["age"]

// Methods
Object.keys(user)
Object.values(user)
```

Python Dictionaries

```
# Create
user = {
    "username": "alex",
    "age": 16
}

# Access
user["username"]
user.get("age")

# Methods
user.keys()
user.values()
```

Key Differences:

JavaScript	Python
`obj.key or obj["key"]`	obj["key"] only
`delete obj.key`	del obj["key"]
`Object.keys(obj)`	obj.keys()
`obj?.key`	obj.get("key")

Common Mistakes

❌ Mistake #1: Using Dot Notation

```
# WRONG
name = user.username  # AttributeError!

# CORRECT
name = user["username"]
```

❌ Mistake #2: Not Using .get()

```
# WRONG (crashes if "age" doesn't exist)
age = user["age"]

# CORRECT (safe)
age = user.get("age", 0)
```

❌ Mistake #3: Modifying While Looping

```
# WRONG
for key in user:
    del user[key]  # RuntimeError!

# CORRECT
for key in list(user.keys()):
    del user[key]
```

❌ Mistake #4: Copying References

```
# WRONG (both point to same dictionary)
user2 = user1

# CORRECT (creates independent copy)
user2 = user1.copy()
```

Quick Examples

Example 1: User Profile

```
# Create user profile
profile = {
    "username": "maria",
    "followers": 5000,
    "verified": True
}

# Access and modify
name = profile.get("username")
profile["followers"] = 5100
profile["posts"] = 42
```

Example 2: Social Media Post

```
# Create post
post = {
    "author": "coder_girl",
    "text": "Learning Python!",
    "likes": 250,
    "comments": 45,
    "shares": 12
}

# Calculate engagement
engagement = (
    post["likes"] +
    post["comments"] +
    post["shares"]
)
print(f"Total engagement: {engagement}")  # 307
```

Example 3: Checking for Keys

```
# Check if key exists
if "email" in user:
    print("Email found!")
else:
    print("No email")
```