Python Dictionaries Methods:

Sure, here are ten more Python dictionary methods with examples:

1. `fromkeys()`: Returns a new dictionary with the specified keys and the specified value.

```python
keys = ("a", "b", "c")
value = 0
my_dict = dict.fromkeys(keys, value)
print(my_dict)  # Output: {"a": 0, "b": 0, "c": 0}
```

2. `setdefault()`: Similar to the `get()` method, but also sets the value for the key if the key is not found.

```python
my_dict = {"a": 1, "b": 2}
value = my_dict.setdefault("c", 3)
print(value)  # Output: 3
print(my_dict)  # Output: {"a": 1, "b": 2, "c": 3}
```

3. `update()`: Updates the dictionary with the key-value pairs from another dictionary or an iterable of key-value pairs.

```python
my_dict = {"a": 1, "b": 2}
new_dict = {"b": 3, "c": 4}
my_dict.update(new_dict)
print(my_dict)  # Output: {"a": 1, "b": 3, "c": 4}
```

4. `keys()`: Returns a view object that contains the keys of the dictionary.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
keys = my_dict.keys()
print(keys)  # Output: dict_keys(['a', 'b', 'c'])
```

5. `values()`: Returns a view object that contains the values of the dictionary.

```python
```

```
my_dict = {"a": 1, "b": 2, "c": 3}
values = my_dict.values()
print(values)  # Output: dict_values([1, 2, 3])
```

6. `items()`: Returns a view object that contains the key-value pairs of the dictionary as tuples.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
items = my_dict.items()
print(items)  # Output: dict_items([('a', 1), ('b', 2), ('c', 3)])
```

7. `pop()`: Removes the key-value pair for the specified key and returns the value.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
value = my_dict.pop("b")
print(value)  # Output: 2
print(my_dict)  # Output: {"a": 1, "c": 3}
```

8. `popitem()`: Removes and returns the last inserted key-value pair.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
key, value = my_dict.popitem()
print(key, value)  # Output: c 3
print(my_dict)  # Output: {"a": 1, "b": 2}
```

9. `clear()`: Removes all the key-value pairs from the dictionary.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
my_dict.clear()
print(my_dict)  # Output: {}
```

10. `copy()`: Returns a shallow copy of the dictionary.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
new_dict = my_dict.copy()
```

```
print(new_dict)
```

Sure, here are 5 more Python dictionary methods with examples:

1. `get()`: Returns the value for the specified key. If the key is not found, it returns the default value (None if not specified).

```python
my_dict = {"a": 1, "b": 2, "c": 3}
value = my_dict.get("b")
print(value)  # Output: 2
```

2. `values()`: Returns a view object that contains the values of the dictionary.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
values = my_dict.values()
print(values)  # Output: dict_values([1, 2, 3])
```

3. `keys()`: Returns a view object that contains the keys of the dictionary.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
keys = my_dict.keys()
print(keys)  # Output: dict_keys(['a', 'b', 'c'])
```

4. `items()`: Returns a view object that contains the key-value pairs of the dictionary as tuples.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
items = my_dict.items()
print(items)  # Output: dict_items([('a', 1), ('b', 2), ('c', 3)])
```

5. `setdefault()`: Returns the value for the specified key. If the key is not found, it inserts the key with the specified value (None if not specified) and returns that value.

```python
my_dict = {"a": 1, "b": 2, "c": 3}
value = my_dict.setdefault("d", 4)
print(value)  # Output: 4
print(my_dict)  # Output: {"a": 1, "b": 2, "c": 3, "d": 4}
```