Manipulating dict

Let us understand how we can manipulate the dicts in Python.

- We can add new key value pairs to `dict` by using typical assignment.
- We can also use assignment operation to update existing key value pair in the `dict`.
- `setdefault` can be used to get the element from the `dict` by using key. If key does not exist, it will update the `dict` with the key passed along with default value.
- `update` can be used to merge a list of pairs (2 tuples) or a `dict` into the `dict`.
- Elements from the dict can be removed using functions like `pop` and `popitem`.
    - `pop` is typically used to remove the element using key.
    - `popitem` is used to remove one of the item (typically last) from the `dict`.

```
d = {'id': 1, 'first_name': 'Scott', 'last_name': 'Tiger', 'amount': 1000.0}
```

```
d['commission_pct'] = 10 # Adding Element
```

```
d['phone_numbers'] = 1234567890
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1000.0,
 'commission_pct': 10,
 'phone_numbers': 1234567890}
```

```
d['amount'] = 1500.0
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1500.0,
 'commission_pct': 10,
 'phone_numbers': 1234567890}
```

```
d = {'id': 1, 'first_name': 'Scott', 'last_name': 'Tiger', 'amount': 1000.0}
```

```
d.setdefault?
```

```
Docstring: D.setdefault(k[,d]) -> D.get(k,d), also set D[k]=d if k not in D
Type:      builtin_function_or_method
```

```
d.setdefault('amount')
```

```
1000.0
```

```
d.setdefault('commission_pct')
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1000.0,
 'commission_pct': None}
```

```
d = {'id': 1, 'first_name': 'Scott', 'last_name': 'Tiger', 'amount': 1000.0}
```

```
d
```

```
{'id': 1, 'first_name': 'Scott', 'last_name': 'Tiger', 'amount': 1000.0}
```

```
d.setdefault('commission_pct', 0)
```

```
0
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1000.0,
 'commission_pct': 0}
```

```
d.setdefault('commission_pct', 100)
```

```
0
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1000.0,
 'commission_pct': 0}
```

```
d.update?
```

```
Docstring:
D.update([E, ]**F) -> None.  Update D from dict/iterable E and F.
If E is present and has a .keys() method, then does:  for k in E: D[k] = E[k]
If E is present and lacks a .keys() method, then does:  for k, v in E: D[k] = v
In either case, this is followed by: for k in F:  D[k] = F[k]
Type:      builtin_function_or_method
```

```
d = {'id': 1}
```

```
d
```

```
{'id': 1}
```

```
d.update({'first_name': 'Donald', 'last_name': 'Duck'})
```

```
d
```

```
{'id': 1, 'first_name': 'Donald', 'last_name': 'Duck'}
```

```
d.update([('amount', 1000.0), ('commission_pct', 10)])
```

```
d
```

```
{'id': 1,
 'first_name': 'Donald',
 'last_name': 'Duck',
 'amount': 1000.0,
 'commission_pct': 10}
```

```
d.update([('amount', 1500.0), ('commission_pct', 5), ('phone_numbers', 1234567890)])
```

```
d
```

```
{'id': 1,
 'first_name': 'Donald',
 'last_name': 'Duck',
 'amount': 1500.0,
 'commission_pct': 5,
 'phone_numbers': 1234567890}
```

```
d = {'id': 1, 'first_name': 'Scott', 'last_name': 'Tiger', 'amount': 1000.0}
```

```
d['commission_pct'] = 10 # Adding Element
```

```
d['phone_numbers'] = 1234567890
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1000.0,
 'commission_pct': 10,
 'phone_numbers': 1234567890}
```

```
d.pop('phone_numbers')
```

```
1234567890
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1000.0,
 'commission_pct': 10}
```

```
d.pop('phone_numbers') # throws KeyError
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-34-430dc980e4cd> in <module>
----> 1 d.pop('phone_numbers') # throws KeyError

KeyError: 'phone_numbers'
```

```
d.pop('phone_numbers', 'No such key exists')
```

```
'No such key exists'
```

```
d.pop?
```

```
Docstring:
D.pop(k[,d]) -> v, remove specified key and return the corresponding value.
If key is not found, d is returned if given, otherwise KeyError is raised
Type:      builtin_function_or_method
```

```
d
```

```
{'id': 1,
 'first_name': 'Scott',
 'last_name': 'Tiger',
 'amount': 1000.0,
 'commission_pct': 10}
```

```
d.popitem?
```

```
Docstring:
D.popitem() -> (k, v), remove and return some (key, value) pair as a
2-tuple; but raise KeyError if D is empty.
Type:      builtin_function_or_method
```

```
d.popitem()
```

```
('commission_pct', 10)
```

```
d
```

```
{'id': 1, 'first_name': 'Scott', 'last_name': 'Tiger', 'amount': 1000.0}
```

© Copyright ITVersity, Inc.