



SWAPPING DICTIONARY  
KEYS AND VALUES |

```
fruit = {"banana": "yellow", "apple": "red", "grape": "purple"}  
  
colors = dict()  
  
for eachFruit in fruit:  
    colors[fruit[eachFruit]] = eachFruit
```

this is my starting dictionary called **fruit**  
it contains key:value pairs of fruit:color




```
fruit = {"banana": "yellow", "apple": "red", "grape": "purple"}
```

dictionary[key] = value

which means that

**fruit["banana"]** refers to **"yellow"**



```
fruit["banana"] = "yellow"  
fruit["apple"] = "red"  
fruit["grape"] = "purple"
```

WHY?

Because if I use the key as an index [] to a dictionary,  
then I get the corresponding value!  
That's the beauty (and the whole point) of dictionaries.

```
fruit = {"banana": "yellow", "apple": "red", "grape": "purple"}
```

```
colors = dict()
```

```
for eachFruit in fruit:  
    colors[fruit[eachFruit]] = eachFruit
```

Empty  
dictionary  
called **colors**,  
where I will be  
putting my  
swapped  
values into

WHY do I need an empty  
dictionary here?

Because Python needs to know  
which, and what kind of, data  
structure you are referring to  
when you assign new values  
later in the for-loop

for **eachFruit** in **fruit** → this means that I am going to loop through every  
key:value pair in my dictionary, where **eachFruit** refers to the key

**colors**[**fruit**[**eachFruit**]] = **eachFruit** → this means that we are  
creating new entries in the **colors** dictionary, where the key  
is equal to **fruit**[**eachFruit**], and the value is equal to  
**eachFruit**

## What happens in my code then?

```
fruit = {"banana": "yellow", "apple": "red", "grape": "purple"}  
  
colors = dict()  
  
for eachFruit in fruit:  
    colors[fruit[eachFruit]] = eachFruit
```

First iteration of my for loop looks like this:

```
colors = {} ← empty  
  
for "banana" in fruit: ← first key in my dictionary  
    colors[fruit["banana"]] = "banana" is "banana"
```

**eachFruit** refers to the key I am currently working with, and since I am in a for-loop, then I process every key in my dictionary in the order they appear  
→ **"banana"** is first

## What happens in my code then?

remember!

```
fruit = {"banana": "yellow", "apple": "red", "grape": "purple"}

colors = dict()

for eachFruit in fruit:
    colors[fruit[eachFruit]] = eachFruit
```

```
fruit["banana"] = "yellow"
fruit["apple"] = "red"
fruit["grape"] = "purple"
```

First iteration of my for loop looks like this:

```
colors = {}

for "banana" in fruit:
    colors[fruit["banana"]] = "banana"
```

MEANING:

```
colors["yellow"] = "banana"
```

I am creating a new entry in my **colors** dictionary, where "yellow" is the key, and "banana" is the value

Because: dictionary[key] = value

## What happens in my code then?

First iteration of my for loop looks like this:

```
colors = {}  
  
for "banana" in fruit:  
    colors[fruit["banana"]] = "banana"
```

colors["yellow"] = "banana"



Second iteration of my for loop looks like this:

```
colors = {"yellow": "banana"}  
  
for "apple" in fruit:  
    colors[fruit["apple"]] = "apple"
```

colors["red"] = "apple"



Third iteration of my for loop looks like this:

```
colors = {"yellow": "banana", "red": "apple"}  
  
for "grape" in fruit:  
    colors[fruit["grape"]] = "grape"
```

colors["purple"] = "grape"



Final result: colors = {"yellow": "banana", "red": "apple", "purple": "grape"}

## Alternative solution:

```
fruit = {"banana": "yellow", "apple": "red", "grape": "purple"}  
  
colors = dict()  
  
for eachFruit, eachColor in fruit.items():  
    colors[eachColor] = eachFruit
```

`fruit.items()` calls key, value pairs from `fruit`  
here, I call keys **eachFruit**, and values **eachColor**