How to Read, Update and Delete Options

Now we’re ready to learn how to read the config file, update its options and even how to delete options. In this case, it’s easier to learn by actually writing some code! Just add the following function to the code that you wrote above.

```python
import configparser
import os

def createConfig(path):
    """
    Create a config file
    """
    config = configparser.ConfigParser()
    config.add_section("Settings")
    config.set("Settings", "font", "Courier")
    config.set("Settings", "font_size", "10")
    config.set("Settings", "font_style", "Normal")
    config.set("Settings", "font_info",
               "You are using %(font)s at %(font_size)s pt")

    with open(path, "w") as config_file:
        config.write(config_file)

def crudConfig(path):
    """
    Create, read, update, delete config
    """
    if not os.path.exists(path):
        createConfig(path)

    config = configparser.ConfigParser()
    config.read(path)

    # read some values from the config
    font = config.get("Settings", "font")
    font_size = config.get("Settings", "font_size")

    # change a value in the config
    config.set("Settings", "font_size", "12")

    # delete a value from the config
    config.remove_option("Settings", "font_style")

    # write changes back to the config file
    with open(path, "w") as config_file:
        config.write(config_file)
```

```
if __name__ == "__main__":


    path = "settings.ini"
    crudConfig(path)
```

This code first checks to see if the path for the config file exists. If it does not, then it uses the createConfig function we created earlier to create it. Next we create a ConfigParser object and pass it the config file path to read. To read an option in your config file, we call our ConfigParser object's **get** method, passing it the section name and the option name. This will return the option's value. If you want to change an option's value, then you use the **set** method, where you pass the section name, the option name and the new value. Finally, you can use the **remove_option** method to remove an option.

In our example code, we change the value of font_size to **12** and we remove the font_style option completely. Then we write the changes back out to disk.

This isn't really a good example though as you should never have a function that does everything like this one does. So let's split it up into a series of functions instead:

```
import configparser
import os

def create_config(path):
    """
    Create a config file
    """
    config = configparser.ConfigParser()
    config.add_section("Settings")
    config.set("Settings", "font", "Courier")
    config.set("Settings", "font_size", "10")
    config.set("Settings", "font_style", "Normal")
    config.set("Settings", "font_info",
               "You are using %(font)s at %(font_size)s pt")

    with open(path, "w") as config_file:
        config.write(config_file)


def get_config(path):
    """
    Returns the config object
    """
    if not os.path.exists(path):
        create_config(path)
    config = configparser.ConfigParser()
    config.read(path)

    return config


def get_setting(path, section, setting):
    """
    Print out a setting
    """
    config = get_config(path)
    value = config.get(section, setting)
    msg = "{section} {setting} is {value}".format(
        section=section, setting=setting, value=value)
    print(msg)
    return value


def update_setting(path, section, setting, value):
    """
    Update a setting
    """
    config = get_config(path)
    config.set(section, setting, value)
    with open(path, "w") as config_file:
        config.write(config_file)


def delete_setting(path, section, setting):
    """
    Delete a setting
    """
    config = get_config(path)
    config.remove_option(section, setting)
    with open(path, "w") as config_file:
        config.write(config_file)



if __name__ == "__main__":
    path = "settings.ini"
    font = get_setting(path, 'Settings', 'font')
    font_size = get_setting(path, 'Settings', 'font_size')

    update_setting(path, "Settings", "font_size", "12")
    font_size = get_setting(path, 'Settings', 'font_size')

    delete_setting(path, "Settings", "font_style")
    font = get_setting(path, 'Settings', 'font_style')
```

This example is heavily refactored compared to the first one. I even went so far as to name the functions following PEP8. Each function should be self-explanatory and self-contained. Instead of putting all the logic into one function, we separate it out into multiple functions and then demonstrate their functionality within the bottom if statement. Now you can import the module and use it yourself.

Please note that this example has the section hard-coded, so you will want to update this example further to make it completely generic.