```
#!/usr/bin/python3
# -- coding: utf-8 -
#**
# * Copyright (c) 2020 Weitian Leung
# *
# * This file is part of pywpsrpc.
# *
# * This file is distributed under the MIT License.
# * See the LICENSE file for details.
# *
#*

import os
import sys

print(sys.path)
import argparse

from pywpsrpc.rpcwpsapi import (createWpsRpcInstance, wpsapi)
from pywpsrpc.common import (S_OK, QtApp)



formats = {
    "doc": wpsapi.wdFormatDocument,
    "docx": wpsapi.wdFormatXMLDocument,
    "rtf": wpsapi.wdFormatRTF,
    "html": wpsapi.wdFormatHTML,
    "pdf": wpsapi.wdFormatPDF,
    "xml": wpsapi.wdFormatXML,
}
```

```
class ConvertException(Exception):

        def __init__(self, text, hr):
                self.text = text
                self.hr = hr

        def __str__(self):
                return """Convert failed:
Details: {}
ErrCode: {}
""".format(self.text, hex(self.hr & 0xFFFFFFFF))



def convert_to(paths, format, abort_on_fails=False):
        hr, rpc = createWpsRpcInstance()
        if hr != S_OK:
                raise ConvertException("Can't create the rpc instance", hr)

        hr, app = rpc.getWpsApplication()
        if hr != S_OK:
                raise ConvertException("Can't get the application", hr)

        # we don't need the gui
        app.Visible = False

        docs = app.Documents

        def _handle_result(hr):
                if abort_on_fails and hr != S_OK:
                        raise ConvertException("convert_file failed", hr)

        for path in paths:
```

```
            abs_path = os.path.realpath(path)
            if os.path.isdir(abs_path):
                files = [(os.path.join(abs_path, f)) for f in os.listdir(abs_path)]
                for file in files:
                    hr = convert_file(file, docs, format)
                    _handle_result(hr)
            else:
                hr = convert_file(abs_path, docs, format)
                _handle_result(hr)

        app.Quit()


def convert_file(file, docs, format):
        hr, doc = docs.Open(file, ReadOnly=True)
        if hr != S_OK:
            return hr

        out_dir = os.path.dirname(os.path.realpath(file)) + "/out"
        os.makedirs(out_dir, exist_ok=True)

        # you have to handle if the new_file already exists
        new_file = out_dir + "/" + os.path.splitext(os.path.basename(file))[0] + "." + format
        ret = doc.SaveAs2(new_file, FileFormat=formats[format])

        # always close the doc
        doc.Close(wpsapi.wdDoNotSaveChanges)

        return ret


def main():
```

```
    parser = argparse.ArgumentParser()
    parser.add_argument("--format", "-f",
                        required=True,
                        metavar="<DOC_TYPE>",
                        choices=["doc", "docx", "rtf", "html", "pdf", "xml"],
                        help="convert to <DOC_TYPE>,")

    parser.add_argument("--abort", "-a",
                        action="store_true",
                        help="abort if one convert fails")

    parser.add_argument("path",
                        metavar="<path>",
                        nargs='+',
                        help="the <path> can be one or more file or folder")

    args = parser.parse_args()

    qApp = QtApp(sys.argv)

    try:
        convert_to(args.path, args.format, args.abort)
    except ConvertException as e:
        print(e)


if __name__ == "__main__":
    main()
```