In [3]:
```
from pathvalidate import sanitize_filepath
```

In [4]:
```
fpath = "/tmp/fi:l*ep\"a?t>h|.t<xt"
```

In [5]:
```
print(f"{fpath} -> {sanitize_filepath(fpath)}\n")
```

```
---------------------------------------------------------------------------
ValidationError                           Traceback (most recent call last)
<ipython-input-5-7d7912d90a3f> in <module>
----> 1 print(f"{fpath} -> {sanitize_filepath(fpath)}\n")

~/anaconda3/lib/python3.8/site-packages/pathvalidate/_filepath.py in sanitize_filepath(file_path, replacement_text,
platform, max_len, check_reserved, normalize)
    418     """
    419
--> 420     return FilePathSanitizer(
    421         platform=platform, max_len=max_len, check_reserved=check_reserved, normalize=normalize
    422     ).sanitize(file_path, replacement_text)

~/anaconda3/lib/python3.8/site-packages/pathvalidate/_filepath.py in sanitize(self, value, replacement_text)
     77             return ""
     78
---> 79         self.__fpath_validator.validate_abspath(value)
     80
     81         unicode_filepath = preprocess(value)

~/anaconda3/lib/python3.8/site-packages/pathvalidate/_filepath.py in validate_abspath(self, value)
    237
    238         if any([self._is_windows(), self._is_universal()]) and is_posix_abs:
--> 239             raise err_object
    240
    241         drive, _tail = ntpath.splitdrive(value)

ValidationError: reason=MALFORMED_ABS_PATH, target-platform=universal, description=an invalid absolute file path (/t
mp/fi:l*ep"a?t>h|.t<xt) for the platform (universal). to avoid the error, specify an appropriate platform correspond
with the path format, or 'auto'.
```

In [6]:
```
print(f"{fpath} -> {sanitize_filepath(fpath, platform='auto')}\n")
```

```
/tmp/fi:l*ep"a?t>h|.t<xt -> /tmp/fi:l*ep"a?t>h|.t<xt
```

In [7]:
```
print(f"{fpath} -> {sanitize_filepath(fpath, platform='Linux')}\n")
```

```
/tmp/fi:l*ep"a?t>h|.t<xt -> /tmp/fi:l*ep"a?t>h|.t<xt
```