Static typing

September 17, 2024

1 Static typing

Python is a strongly typed and dynamically typed programming language.

Dynamic typing means that variables can refer to different types of objects at different times and that collections can contain references to different types of objects, and that these types can vary at runtime.

However, these properties are rarely used. Most variables don't change type over the lifetime of a program and most collections have uniform types.

For this reason it makes sense to include *type hints* in Python programs for readability reasons and for improved code quality. We eliminate a whole category of bugs if we can make a static evaluation of types in the places where we use operators. Please note that in Python, function calls are operators. It turns out that adding type hints to function signatures gives us the best return on investment.

Python implements **incremental typing**. This means that code without type hints is not checked. Only the places where we have added hints get checked. The various checking tools also make **type inferences** where types can be calculated.

There are several tools that do **static typechecking**. The best ones are currently:

- **pyright** - also known as **pylance**, when used as a pluging to VSCode -created by Microsoft
- **mypy** - created by Dropbox when Guido van Rossum worked there

https://mypy.readthedocs.io/en/stable/cheat_sheet_py3.html

[4]:
```
from typing import reveal_type
i = 1
reveal_type(i)
```

```
Runtime type is 'int'
```

[4]: 1

[ ]: