# Python Types and Type Hints

## Usage at octopusenergy.com

Daniel Feldroy

# A Typical Octopus Python Function

```python
def convert_to_green_energy(email, esi_id, conditions):
    """
        This function converts a Texas address in the ERCOT region from
        using fossil fuels to renweable energy from wind and solar.

        Parameters:

            email: String that is a valid email address
            esi_id: Integer for interacting with ERCOT
            conditions: The quirks of the property and smart meter that have
                to be tracked.

        Returns:
            dict: Contains whether or not it succeeded, start date, money saved,
            and other values
    """
    # Much logic goes here
    # Much logic goes here
    # Much logic goes here
    # Much logic goes here
    return {
        'success': success,
        'transfer_date': datetime.datetime(value),
        'money_saved': lots_of_dollars
    }
```

# The Same Octopus Python Function With Type Hints

```python
def convert_to_green_energy(email: str, esi_id: int, conditions: list) -> dict:
    """
        This function converts a Texas address in the ERCOT region from
        using fossil fuels to renweable energy from wind and solar.
    """
    # Much logic goes here
    # Much logic goes here
    # Much logic goes here
    # Much logic goes here
    return {
        'success': success,
        'transfer_date': datetime.datetime(value),
        'money_saved': lots_of_dollars
    }
```

## What just happened?!

What just happened?!

Answer: We leaned on Python's typing system.

## What is Python's typing system?

## Let's start with types.

# Type

*noun, often attributive*

- a particular kind, class, or group
- something distinguishable as a variety

Source: Merriam-Webster

# Types in Python

- `str`: A sequence of unicode characters, called a string
- `int`: Whole numbers, called integers in math class
- `list`: A list of things
- `dict`: A set of keys and their values

## Examples of Types in Python

```
# str
name = "Uma Roy Greenfeld"

# her age as a whole number
age = 2

# A list of her favorite things
favorites = ['ducks', 'blueberries', 'singing']

# A dictionary of toy types and their quantities
toys = {
    'rubber ducks': 9,
    'plush animals': 24,
    'books'
    'legos': 9581204829450
}
```

## How to break Python with types

```
# Can't combine str and int
name + age

# Can't combine str and list
name + favorites
```

This typing is a good thing. Prevents us from fitting square pegs in round holes.

## Example of Breaking Python Code by ignoring types

```
convert_to_green_energy(
    # a list of favorites is not email
    email=favorites,
    # ERCOT ID is not email, it is a 17 or 22 length string of digits
    esi_id=email,
    # A list of conditions is not a name
    conditions=name
)
```

## These bullets are all true

- Most coders barely read docstrings or docs.
- Walls of text are often impenetrable.
- For basic comprehension, concise is better than verbose.

## What's easier to comprehend?

## A single line of concise information

```python
def convert_to_green_energy(email: str, esi_id: int, conditions: list) -> dict:
```

## A wall of text saying the same thing

```python
"""
Parameters:

    email:str String that is a valid email address
    esi_id:int Integer for interacting with ERCOT
    conditions:list The quirks of the property and smart meter that have
        to be tracked.

Returns:
    dict: Contains whether or not it succeeded, start date, money saved,
    and other values
"""
```

## Summary of Benefits

```
def convert_to_green_energy(email: str, esi_id: int, conditions: list) -> dict:
```

- Much more concise: 1 line vs 11
- IDEs and markdown renderers color the code for increased clarity
- Easier for IDEs to provide lookups
- I can never, ever remember the official docstring format for params and returns. I promise I got it wrong.

## Python Type Hints are awesome!

## What about optional arguments?

```python
def sell_electricity_back_to_grid(
    email: str,
    kilowatts: int = 0) -> float:
    """
    Customers with solar panels sell back to the grid.
    """
    # Selling logic here
    # Selling logic here
    # Selling logic here
    return credit_earned
```

*If kilowatts isn't supplied we default to 0.*

## Notes about docstrings and comments

- Type hints don't take away the need for docs and comments
- They allow docs and comments to focus on logic and purpose
- The decrease in docs volume makes them concise
- Concise (not terse) docs tend to have more clarity

## Amazing Type Hint Libraries

- FastAPI
- Typer

## About Octopus Energy

## About Octopus Energy I

- Saving the planet by selling renewables cheaply
- Home office is in the UK - octopus.energy
- Just started operations in the US - octopusenergy.com

## About Octopus Energy II

- At great expense we covered our customer's outrageous bills during winter storm
- We'll be hiring in Houston soon!

## About Me

- Engineering manager at Octopus Energy
- Author
    - Two Scoops of Django
    - A Wedge of Djangoddg
- Open source maintainer
- aka Daniel Roy Greenfeld aka pydanny

Questions?