Python Comprehensions

List comprehensions (or just comprehensions) are one of the most powerful tools python has for quickly creating lists; you can use them to generate mathematical series, extract relevant fields from a group of objects, or transform your data quickly from one class/structure/schema to another. This talk will be an introduction to the syntax of comprehensions and examples of how to use them to do all kinds of amazing things in just a line of code.

Purpose

Shorthand for transformations and/or iterations.

Basic idea

`[ value for item in list ]` -> new list

Example: computing squares

```
squares = [i**2 for i in range(1, 26)]
```

What it does:

- for every number from 1 to 25...
- compute the square of that number
- store it in the list

Same code in a for loop:

```
squares = []
for i in range(1, 26):
    squares.append(i)
```

Example: extracting field from list of objects

```
people = (some list, probably read from a database or a JSON file or whatever)
names = [p.name for p in people]
# iterate over myList
# for each object -> assign it to a local variable called 'p'
# call the .name getter for that object
```

Same code in a for loop:

```
names = []
for p in names:
    names.append(p.name)
```

Advantages of comprehensions

- fewer lines of code
- easier to see purpose of all variables
- easier to see corner cases

Advanced uses

Only include people who match condition

```
[p.name for p in people if p.username.startswith('a')]
```

Include all people but use alternate value for some

```
[p.name if p.name else 'unknown' for p in people]
```

Combine both conditions

```
[p.name if p.name else 'unknown' for p in people if p.username.startswith('a')]
```

Other result types (besides list)

- set (unordered): `{x for x in numbers}`
- dictionary: `{p.id:p for p in people}`

Examples

Create lookup table of people by ID

```
peopleById = { p.id:p for p in people }
```

Create a lookup table of people names by ID

```
nameById = { p.id:p.name for p in people }
```

Get a sorted list of usernames

```
usernames = sorted([p.username for p in people])
```

Get all people with IDs that end in a '1'

```
users = [p for p in people if p.id%10==1]
```

Resources

- Python: List Comprehensions
- Python List Comprehensions: Explained Visually