





Method syntax

Michael Ernst

UW CSE 140

Digression: two meanings of period (.)

- **Namespace management:** `math.sin(math.pi)`
 - Purpose: avoid polluting the global environment frame with too many names
 - Example problem:
 - `abs` is in the global namespace
 - When writing a program about bodybuilding, you might use `abs` for “abdominals”
 - Suddenly, all code that depends on the built-in `abs` fails!
 - Example: `draw()` function for pictures, for cowboys, and for curtains
 - Solution: A different namespace or module
- 
- 
- 
- 
- **“Method syntax” for a function call:** `mylist.append(new_elt)`
 - `append()` takes *two* arguments
 - First argument may appear before a period
 - These are *distinct* meanings of a single token, “.”
 - Determine which by whether the expression before the dot is a type

Method call syntax

- Ordinary function call: `fn(arg1, arg2, arg3)`
 - Any Python function can be invoked this way
- “Method syntax”: `arg1.fn(arg2, arg3)`
 - First argument may appear before a period
 - Some Python functions can be invoked this way
 - Only works for a function defined in a type’s namespace
 - Such a function is called a “method”
 - We will not learn how to create methods in CSE 140, but we will use them

```
nums = [3, 1, 4]
```

```
nums.append(1)
```

```
list.append(nums, 5)
```

```
append(nums, 9)
```

```
list.append
```

```
nums.append
```

`nums` is *not* a type, so this
is method call syntax

`list` is a type, so this
is namespace lookup

```
# append is defined in the list namespace
```

```
# NameError: name 'append' is not defined
```

```
# a function that takes 2 arguments
```

```
# the same function (1 arg is already given)
```

A function is invoked differently, depending on where it is defined

```
nums.index(4)
```

```
len(nums)
```

```
nums.len()    # AttributeError: 'list' object has no attribute 'len'
```

Reason: **len** is defined at the top level, not in the **list** namespace

```
sorted(nums)
```

```
print nums
```

```
nums.sort()
```

```
sort(nums)    # NameError: name 'sort' is not defined
```

Reason: **sort** is defined in the **list** namespace, not at the top level