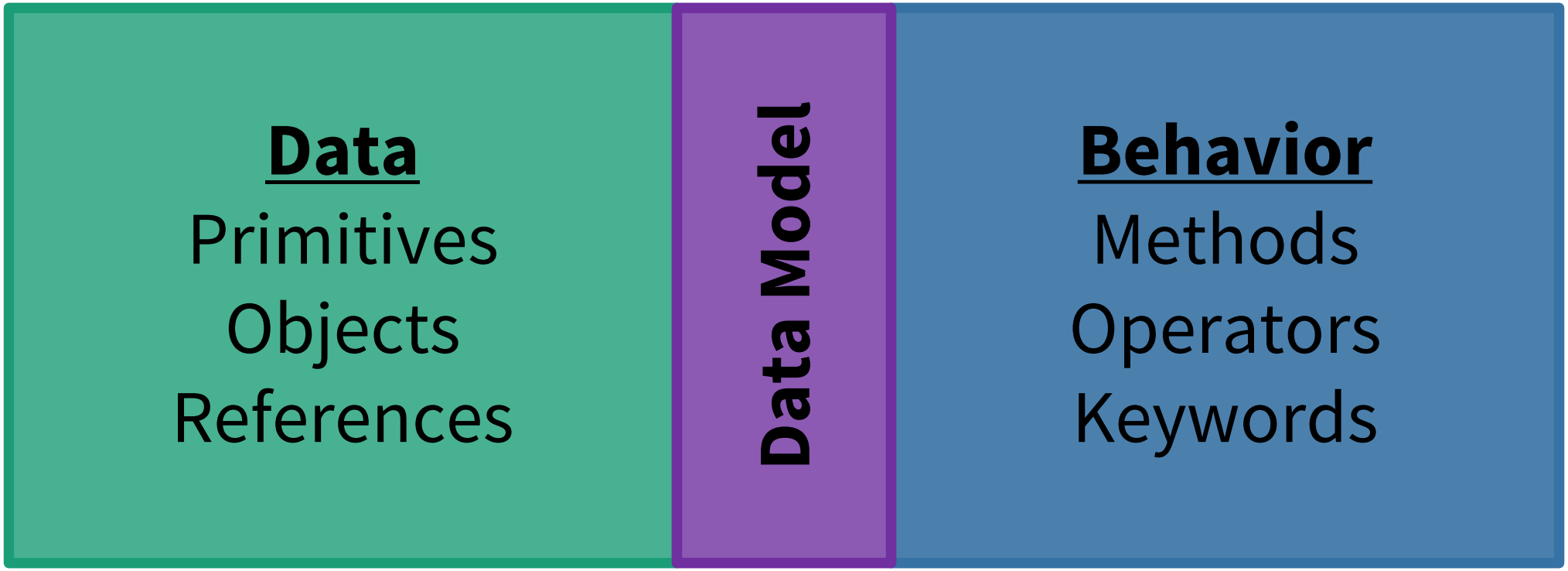
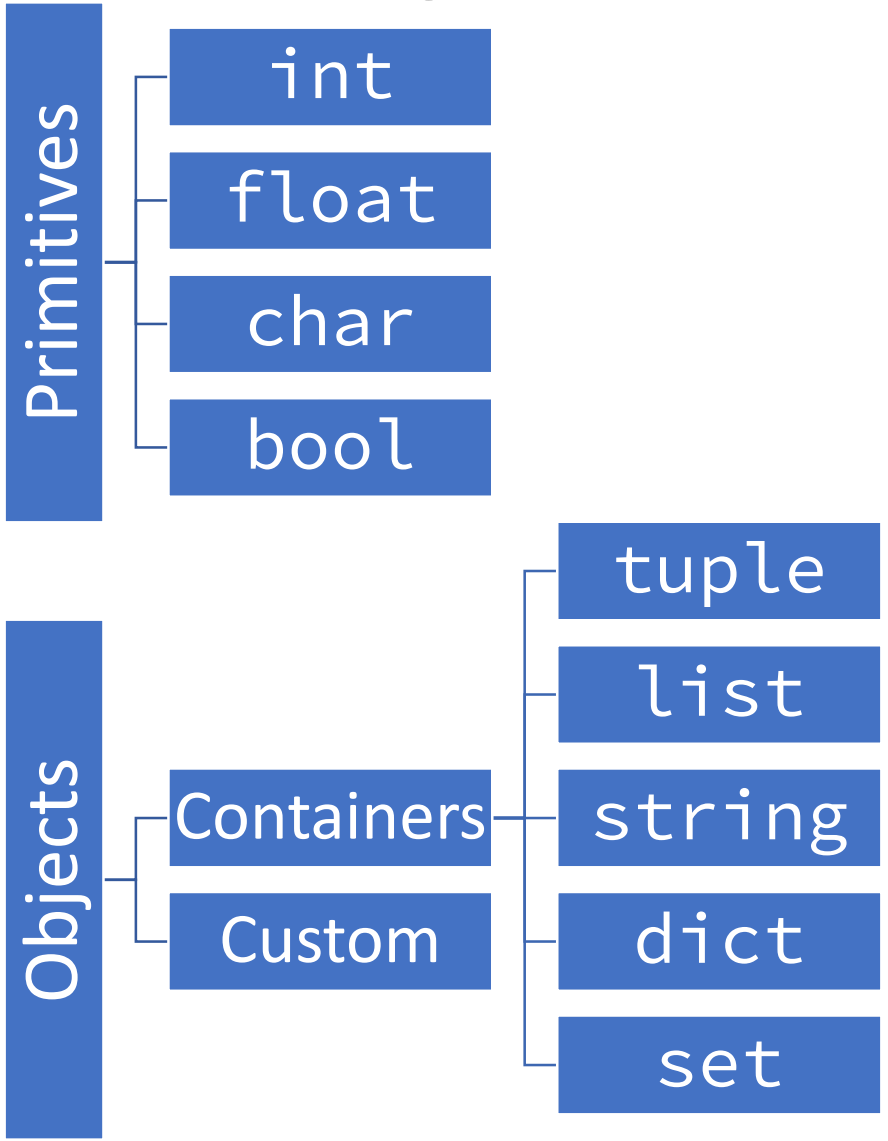
The Python Data Model

Josh Karpel

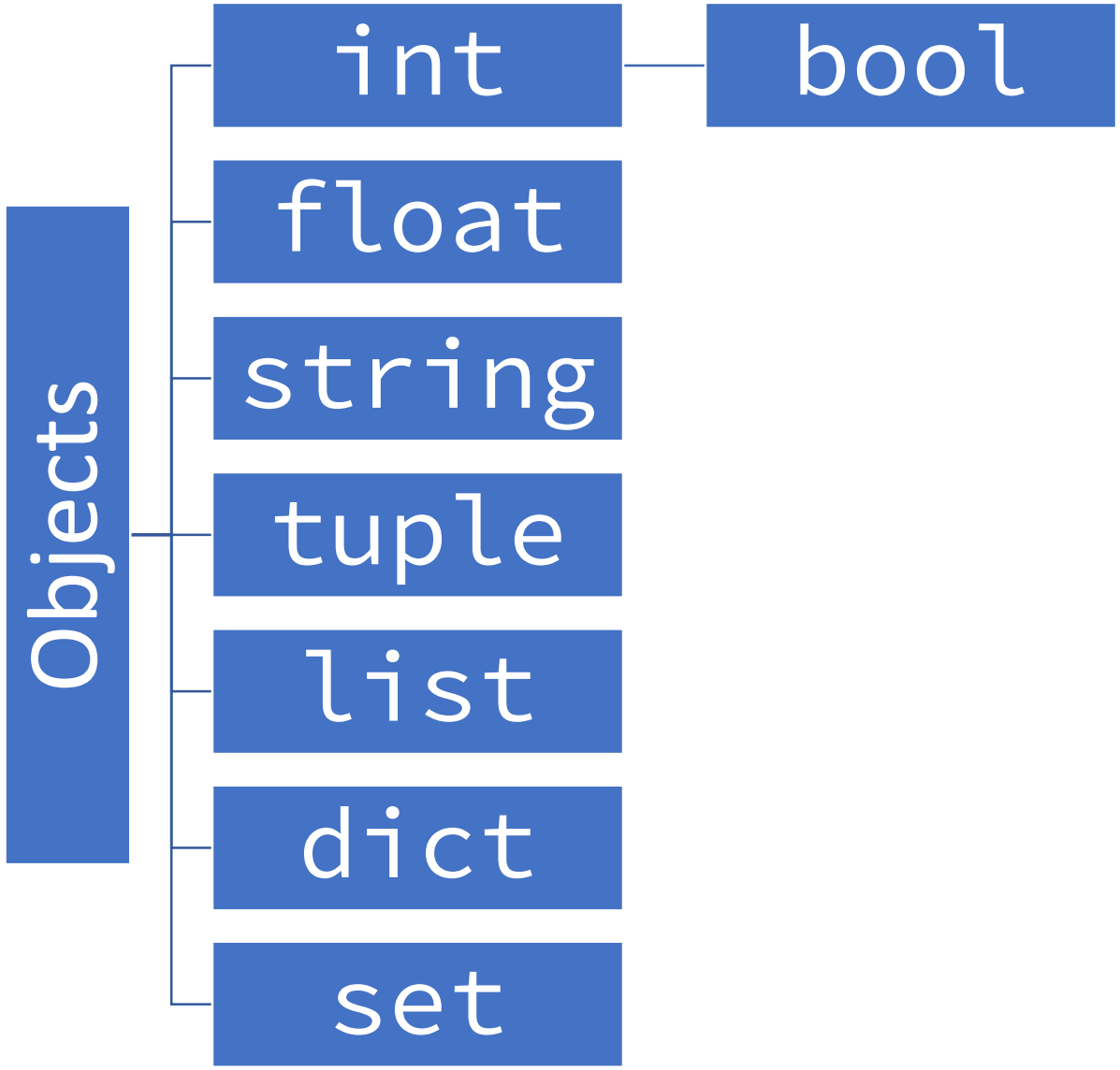
The Python Data Model?

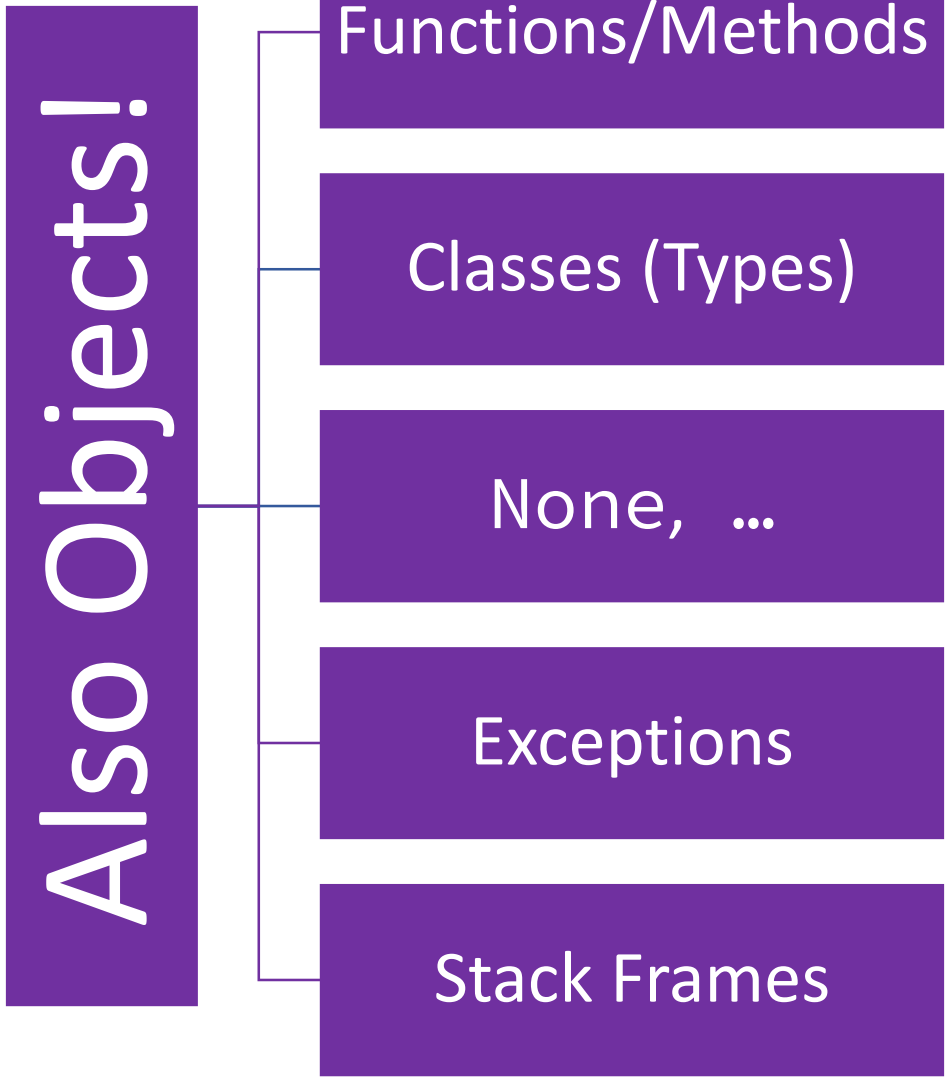


An Accounting of All Things…



In Python, Everything Is An Object





Many things that look like

builtins

are actually

protocols

The Python Data Model: Object Behavior is Defined by Special Methods

Some "Builtins"	Underlying Protocol Methods
+, -, *, etc.	__add__, __sub__, __mul__, etc.
[ ]	__getitem__, __setitem__
( )	__call__
.	__getattr__, __setattr__
&, |	__and__, __or__
with <object>:	__enter__, __exit__

Python "Interfaces"

Functions for sequences

`random.`**`choice`**`(`*seq*`)`

Return a random element from the non-empty sequence *seq*. If *seq* is empty, raises `IndexError`.

Python "Interfaces"

```
def choice(self, seq):
    """Choose a random element from a non-empty sequence."""
    try:
        i = self._randbelow(len(seq))
    except ValueError:
        raise IndexError('Cannot choose from an empty sequence') from None
    return seq[i]
```

Python "Interfaces"

```
def choice(self, seq):
    """Choose a random element from a non-empty sequence."""
    try:
        i = self._randbelow(len(seq))
    except ValueError:
        raise IndexError('Cannot choose from an empty sequence') from None
    return seq[i]
```

Example:
A **Hand** of **Cards**

Why Bother?

The Zen of Josh

Names are Powerful

Abstraction creates Simplicity

Authors are Users Too

Python's all-encompassing

Data Model is a perfect tool for

implementing the Zen of Josh

Example:
Polynomials

Example:
Polynomials

Wrapping the underlying data structure made it more useful and intuitive

Example:
Printing Colored Text

Example:
Printing Colored Text

Abstraction works on
both data and behavior

The Zen of the Zen of Josh

Names are Powerful

… but a wheel is a wheel is a wheel

Abstraction creates Simplicity

… but creates complexity elsewhere

Authors are Users Too

… but outside users don't care

Further Resources

The Bible

The Python Data Model

The Other Bible

Fluent Python

Talks

Beyond PEP 8

Transforming Code into Beautiful, Idiomatic Python

Functional Core, Imperative Shell