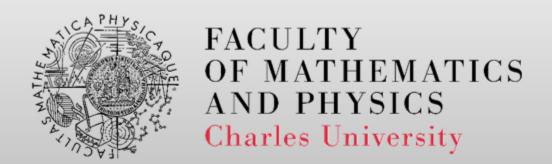
NPRG065: Programming in Python Lecture 11

http://d3s.mff.cuni.cz





Tomas Bures
Jan Kofron

{bures,kofron}@d3s.mff.cuni.cz

- __setattr__(self, name, value)
 - called when an attribute assignment is attempted
- getattr (self, name)
 - called when the default attribute access fails with an AttributeError
 - if an attribute already has a value, __getattr__() is not used
- __delattr__(self, name)
 - like __setattr__() but for attribute deletion instead of assignment
- dir (self)
 - called when dir() is called on the object
 - a sequence must be returned

See attrs.py



• dir(object)

- if the object has a method named __dir__(), this method will be called
- if the object does not provide __dir__(), the function tries to gather information from the object's __dict__ attribute, if defined, and from its type object

• __dict_

- a dictionary used to store an object's attributes
- created automatically



3

- Why use __xxxattr__()
 - creating immutable objects (with __slots__)
 - lazy creation of attribute values
 - own property-like behavior but in a single method
 - creating attributes when setting values to other attributes
 - **.**.



- __slots__
 - a class variable that can be assigned a string, iterable, or sequence of strings with variable names used by instances
 - reserves space for the declared variables
 - prevents the automatic creation of ___dict___
 - i.e., no other than declared variables can be created
 - plus objects with slots are smaller and faster
 - there is no dynamic dict

See slots.py



- getattribute (self, name)
 - called unconditionally to attribute accesses
 - default implementation locates value in __dict__ or __slots__
 - if the class also defines __getattr__(), the latter will not be called unless __getattribute__() either calls it explicitly or raises an AttributeError
 - to access other attributes from __getattribute__, call the base class method (to avoid recursion)
 - object.__getattribute__(self, name)
 - usages
 - preventing access to attributes
 - inventing new attributes (like with __getattr__ but without look for existing attributes)
 - rarely used

See attribute.py



Function decorators

```
@decorate
def function():
    pass
```

- decorator ~ a function modifying a function to create new function
 - code above is equivalent to

```
def function():
    pass
function = decorate( function )
```

See fdecorators.py



Function decorators

- many predefined decorators
 - @property, @staticmethod
 - we already know them
 - functools module
 - many useful decorators (not only for defining other decorators)



Class decorators

- Similar to function decorators
- A function receiving a class object as an argument and returning a class object as a result
- Less commonly used than function decorators

See cdecorators.py





