Python Easter Egg Hunt

- Anti-gravity
- Braces
- Geohashing
- Spam, Ham, and Eggs
- Hello World
- Infinity
- PEG Parser
- The Diamond Operator
- The Zen of Python

What is This About?

Python has its fair share of hidden surprises, commonly known as Easter eggs. From clever jokes to secret messages, these little mysteries are often meant to be discovered by curious geeks like you!

How to Use This Guide?

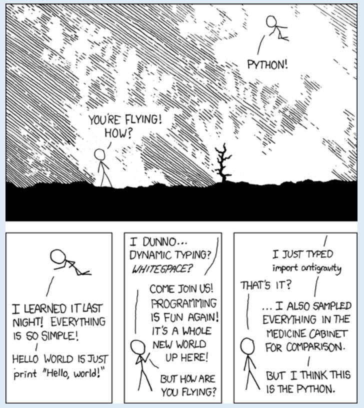
Here's a guide to help you in your Easter egg hunt. On the left-hand side, you'll find the eggs to uncover. By moving to the next slide, you'll reveal a hint that should give you a clue on where to look. Once you think you've found the egg, move to the subsequent slide and check your solution against the answer key. Rinse and repeat.

Good luck!

Python Easter Egg Hunt

- 🐣 **Anti-gravity**
- 🥚 Braces
- 🥚 Geohashing
- 🥚 Spam, Ham, and Eggs
- 🥚 Hello World
- 🥚 Infinity
- 🥚 PEG Parser
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Randall Munroe, the creator of the popular xkcd webcomic, often features Python in his cartoons. One of them is a humorous tribute to the simplicity and elegance of the Python programming language, showing a character who apparently achieves anti-gravity by using one of the "batteries" included in Python:


YOU'RE FLYING! HOW?
PYTHON!
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

https://xkcd.com/353/

Hint: Can you guess what battery the comic is referring to?

Python Easter Egg Hunt

- 🐥 **Anti-gravity**
- 🥚 Braces
- 🥚 Geohashing
- 🥚 Spam, Ham, and Eggs
- 🥚 Hello World
- 🥚 Infinity
- 🥚 PEG Parser
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Solution

Run the `antigravity` module with Python:

```
$ python -m antigravity
```

Or, import it in a Python REPL:

```
>>> import antigravity
```

This will open the xkcd comic in your default web browser.

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐣 **Braces**
- 🥚 [illegible]
- 🥚 [illegible]
- 🥚 [illegible]
- 🥚 [illegible]
- 🥚 [illegible]
- 🥚 [illegible]
- 🥚 [illegible]

Many programmers coming to Python from languages like C, C++, or Java may be accustomed to using curly braces `{}` for defining blocks of code:

```
if (age > 18) {
    printf("You can vote\n");
}
```

On the other hand, Python uses indentation to define code blocks:

```
if age > 18:
    print("You can vote")
```

Hint: Remember how you enable experimental features or those planned for upcoming Python releases.

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 **Braces**
- 🥚 Geohashing
- 🥚 Spam, Ham, and Eggs
- 🥚 Hello World
- 🥚 Infinity
- 🥚 PEG Parser
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Solution

Import `braces` from the `__future__` module:

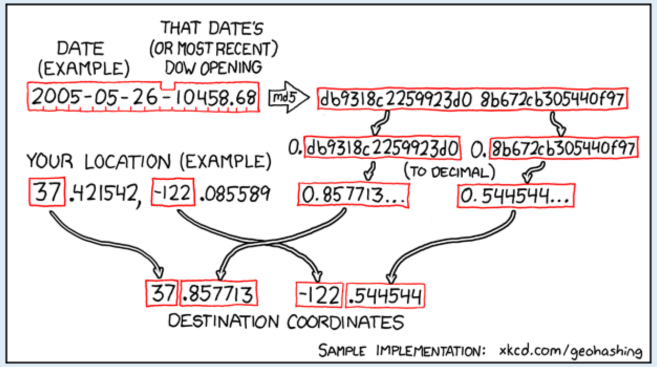
```
>>> from __future__ import braces
  File "<stdin>", line 1
SyntaxError: not a chance
```

As you can see, Python promises never to support this feature, staying true to its design philosophy of using indentation for code blocks.

Python Easter Egg Hunt

- 🐣 Anti-gravity
- 🐣 Braces
- 🐣 **Geohashing**
- 🥚 Spam, Ham, and Eggs
- 🥚 Hello World
- 🥚 Infinity
- 🥚 PEG Parser
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Geohashing is an algorithm revealed by xkcd in another comic, which you can use for geocaching or other treasure-hunting activities. The adventure begins by obtaining pseudo-random geographic coordinates based on the current date of the year, the corresponding stock market data, and the MD5 hashing function:



https://xkcd.com/426/

Hint: You can always introspect the content of Python batteries, even if they're deliberately undocumented.

Python Easter Egg Hunt

- Anti-gravity
- Braces
- **Geohashing**
- Spam, Ham, and Eggs
- Hello World
- Infinity
- PEG Parser
- The Diamond Operator
- The Zen of Python

Solution

The `antigravity` module defines a `geohash()` function, which takes the initial latitude and longitude, as well as a byte string with the current date in ISO 8601 format followed by the stock market index:

```
>>> from antigravity import geohash
>>> geohash(37.421542, -122.085589, b"2005-05-26-10458.68")
37.857713 -122.544543
```

The function prints a new set of seemingly random geographic coordinates within one degree in the horizontal and vertical directions. It implements the geohashing algorithm depicted in the xkcd comic.

Python Easter Egg Hunt

- Anti-gravity
- Braces
- Geohashing
- **Spam, Ham, and Eggs**
- Hello World
- Infinity
- PEG Parser
- The Diamond Operator
- The Zen of Python

In Python, *spam*, *ham*, and *eggs* are placeholder names often used in examples throughout the official documentation. These metasyntactic variables serve a similar purpose as *foo*, *bar*, or *baz* in other programming languages.

Python's quirky names were humorously inspired by a famous Monthy Python sketch involving a cafe that serves food items mainly containing Spam.

Hint: See if you can find one of these names in Python's source code on GitHub.

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 **Spam, Ham, and Eggs**
- 🥚 Hello World
- 🥚 Infinity
- 🥚 PEG Parser
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Solution

A few unit tests in Python take advantage of the `__phello__` package hidden in the standard library, which contains submodules with the familiar names:

```
>>> from __phello__ import spam
>>> from __phello__ import ham
>>> from __phello__.ham import eggs
```

These modules don't contain any meaningful code other than a minimal implementation of the "Hello, World!" program:

```
>>> import __phello__
>>> __phello__.main()
Hello world!
```

Real Python

Python Easter Egg Hunt

- Anti-gravity
- Braces
- Geohashing
- Spam, Ham, and Eggs
- **Hello World**
- Infinity
- PEG Parser
- The Diamond Operator
- The Zen of Python

Writing a “Hello, World!” program has become the first thing that most programmers do when they start learning a new language. Implementing it in Python is fairly straightforward, as it boils down to one line of code:

```
print("Hello, World!")
```

Still, there’s an even simpler way buried in the Python source code!

Hint: Check if Python’s unit tests leverage other suspiciously-looking modules different than the `__phello__` package that you just saw.

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 **Hello World**
- 🥚 Infinity
- 🥚 PEG Parser
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Solution

There's a `__hello__` module, which you can run directly from the terminal:

```
$ python -m __hello__
Hello world!
```

Now, can your programming language do that? 😎

Python Easter Egg Hunt

- Anti-gravity
- Braces
- Geohashing
- Spam, Ham, and Eggs
- Hello World
- **Infinity**
- PEG Parser
- The Diamond Operator
- The Zen of Python

The hash() of a whole number in Python is equal to the respective numeric value. On the other hand, the hashing algorithm maps fractions to some arbitrary integers:

```
>>> hash(4)
4
>>> hash(4.0)
4
>>> hash(4.0 + 0.0j)
4

>>> hash(3.14)
322818021289917443
>>> hash(322818021289917443)
322818021289917443
```

Hint: What about the few sentinels defined in the IEEE 754 standard, which don't have a numeric value?

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 Hello World
- 🐥 **Infinity**
- 🥚 PEG Parser
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Solution

The hash value of the floating-point positive and negative infinity consists of the first six digits of π:

```
>>> hash(float("inf"))
314159

>>> hash(float("-inf"))
-314159
```

You can see why when you inspect the `_PyHASH_INF` constant in Python’s source code.

Python Easter Egg Hunt

- Anti-gravity
- Braces
- Geohashing
- Spam, Ham, and Eggs
- Hello World
- Infinity
- **PEG Parser**
- The Diamond Operator
- The Zen of Python

Python 3.9 introduced a more powerful parser for its source code, which is based on the Parsing Expression Grammar (PEG).

While mostly invisible to the end user, the PEG parser was a major milestone, simplifying the parsing logic and allowing the language's syntax to evolve more flexibly.

The Python 3.9 release bundles both the old LL(1) and the new PEG parser, letting you toggle between them. By default, it runs the new PEG parser, but you can optionally use the old parser with this command-line flag:

```
$ python3.9 -X oldparser
```

Hint: Check if any new keywords were introduced along with the PEG parser.

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 Hello World
- 🐥 Infinity
- 🐥 **PEG Parser**
- 🥚 The Diamond Operator
- 🥚 The Zen of Python

Solution

There had been a new `__peg_parser__` keyword in Python 3.9, which was removed from subsequent releases:

```
>>> from keyword import kwlist
>>> kwlist
['False', 'None', 'True', '__peg_parser__', ..., 'yield']
```

It raised a syntax error with a secret message when you ran the interpreter with the PEG parser enabled:

```
>>> __peg_parser__
  File "<stdin>", line 1
    __peg_parser__
    ^
SyntaxError: You found it!
```

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 Hello World
- 🐥 Infinity
- 🐥 PEG Parser
- 🐣 **The Diamond Operator**
- 🥚 The Zen of Python

The legacy Python 2 had two alternate spellings for the inequality test or the “not-equal-to” operator. One of them consisted of the exclamation mark followed by an equals sign (`!=`), which was familiar to C developers. The thinking here was that these symbols resemble a crossed equals sign (`≠`). The other spelling was more intuitive to Pascal developers, who were used to the diamond operator (`<>`):

```
>>> if 1 <> 2:
...     print "This is Python 2"
```

This operator is no longer available in Python 3, but maybe there’s some way to import it from the `__past__`?

Hint: Review the Python Enhancement Proposal (PEP) documents to determine if there were attempts at bringing back the old syntax in Python 3.

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 Hello World
- 🐥 Infinity
- 🐥 PEG Parser
- 🐥 **The Diamond Operator**
- 🥚 The Zen of Python

Solution

In 2009 on April 1st—or 4/01 in the US date format—PEP 401 was accepted as an April Fools' Day joke. It claimed the retirement of Guido van Rossum as the Benevolent Dictator for Life (BDFL). Simultaneously, Barry Warsaw, also known as the Friendly Language Uncle For Life (FLUFL), would become his successor and introduce controversial changes, including the reinstatement of the diamond operator:

```
>>> from __future__ import barry_as_FLUFL

>>> 1 != 2
  File "<stdin>", line 1
    1 != 2
      ^^
SyntaxError: with Barry as BDFL, use '<>' instead of '!='

>>> 1 <> 2
True
```

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 Hello World
- 🐥 Infinity
- 🐥 PEG Parser
- 🐥 The Diamond Operator
- 🐣 **The Zen of Python**

Hint: By now, you should already know what *this* is.

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 Hello World
- 🐥 Infinity
- 🐥 PEG Parser
- 🐥 The Diamond Operator
- 🐥 **The Zen of Python**

Solution

Imoprting `this` reveals the Zen of Python by Tim Peters:

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Python Easter Egg Hunt

- 🐥 Anti-gravity
- 🐥 Braces
- 🐥 Geohashing
- 🐥 Spam, Ham, and Eggs
- 🐥 Hello World
- 🐥 Infinity
- 🐥 PEG Parser
- 🐥 The Diamond Operator
- 🐥 The Zen of Python