

# Introduction to Python Dictionaries

March 13, 2012

# The Big Picture

## Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

## Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

# Word Frequency: Inputs and Outputs

The cat had a hat. The cat sat on the hat.

I want to write a `wordFreq` function

# Word Frequency: Inputs and Outputs

The cat had a hat. The cat sat on the hat.

I want to write a `wordFreq` function

- What is the input to `wordFreq`?

# Word Frequency: Inputs and Outputs

The cat had a hat. The cat sat on the hat.

I want to write a `wordFreq` function

- What is the input to `wordFreq`?
- What is the output of `wordFreq`?

# Word Frequency: Inputs and Outputs

The cat had a hat. The cat sat on the hat.

I want to write a `wordFreq` function

- What is the input to `wordFreq`?
- What is the output of `wordFreq`?

Word	Freq.
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

# Word Frequency: Inputs and Outputs

The cat had a hat. The cat sat on the hat.

I want to write a `wordFreq` function

- What is the input to `wordFreq`?
- What is the output of `wordFreq`?
- *We could* do this with a list. How?

Word	Freq.
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

# The Big Picture

## Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

## Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

# A New Data Structure

- *A Data Structure* is simply a way to store information.
  - **Lists** are a type of data structure
    - We can have **lists** of integers, floats, strings, booleans, or any combination.
    - Organized **linearly** (indexed by a range of integers)

# A New Data Structure

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

# A New Data Structure

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Associate each **word** with the **frequency**.

# A New Data Structure

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Associate each **word** with the **frequency**.

Keys

Values

# A New Data Structure

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Associate each **word** with the **frequency**.

Keys

Values

Key-Value Pairs

Key	Value
'the'	3
'cat'	2

# A New Data Structure: Dictionaries

Keys and values can be *any* type or data structure

Key Type	Value Type	Example Key	Example Value

# A New Data Structure: Dictionaries

Keys and values can be *any* type or data structure

Key Type	Value Type	Example Key	Example Value
String	Integer	`the`	3
String	Integer	`cat`	2

# A New Data Structure: Dictionaries

Keys and values can be *any* type or data structure

Key Type	Value Type	Example Key	Example Value
String	Integer	`the`	3
String	Integer	`cat`	2
String	String	`Alice`	`401-111-1111`
String	String	`Bob`	`401-222-2222`

# A New Data Structure: Dictionaries

Keys and values can be *any* type or data structure

Key Type	Value Type	Example Key	Example Value
String	Integer	`the`	3
String	Integer	`cat`	2
String	String	`Alice`	`401-111-1111`
String	String	`Bob`	`401-222-2222`
Integer	String	1	`one`
Integer	String	2	`two`

# A New Data Structure: Dictionaries

Keys and values can be *any* type or data structure

Key Type	Value Type	Example Key	Example Value
String	Integer	`the`	3
String	Integer	`cat`	2
String	String	`Alice`	`401-111-1111`
String	String	`Bob`	`401-222-2222`
Integer	String	1	`one`
Integer	String	2	`two`
Float	List	3.0	['t', 'h', 'r', 'e', 'e']

# A New Data Structure: Dictionaries

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq = {}  
>>> freq  
{ }  
>>>
```

Initialize a Dictionary

# A New Data Structure: Dictionaries

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq = {}
>>> freq
{}
>>> freq['the'] = 3
>>> freq
{'the': 3}
>>>
```

Initialize a Dictionary

Key = 'the'  
Value = 3

# A New Data Structure: Dictionaries

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq = {}
>>> freq
{}
>>> freq['the'] = 3
>>> freq
{'the': 3}
>>> freq['cat'] = 2
>>> freq
{'the': 3, 'cat': 2}
>>>
```

Initialize a Dictionary

Key = 'the'  
Value = 3

Key = 'cat'  
Value = 2

# A New Data Structure: Dictionaries

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq2 = {'the':3, 'cat':2}
>>> freq2
{'the': 3, 'cat': 2}
>>>
```

Initialize a dictionary  
with two key-value pairs

# A New Data Structure: Dictionaries

The cat had a hat. The cat sat on the hat.

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq2 = {'the':3, 'cat':2}
>>> freq2
{'the': 3, 'cat': 2}
>>>
>>> freq2['cat']
2
>>> freq2['the']
3
```

Retrieve a value  
using the key

# The Big Picture

## Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

## Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

# Python Dictionaries

Do Task 1

Function (All operate on Dictionaries)	Input	Output	Example
<code>keys()</code>	None	List of keys	<pre>&gt;&gt;&gt; freq2.keys() ['the', 'cat']</pre>
<code>values()</code>	None	List of values	<pre>&gt;&gt;&gt; freq2.values() [3, 2]</pre>
<code>has_key(&lt;key&gt;)</code>	Key	Boolean	<pre>&gt;&gt;&gt; freq2.has_key('missing') False</pre>
<code>&lt;key&gt; in &lt;dict&gt;</code>	(means same as above)		<pre>&gt;&gt;&gt; 'the' in freq2 True</pre>
<code>del(&lt;dict&gt;[&lt;key&gt;])</code>	Dict. Entry	None	<pre>&gt;&gt;&gt; del(freq2['cat'])</pre>

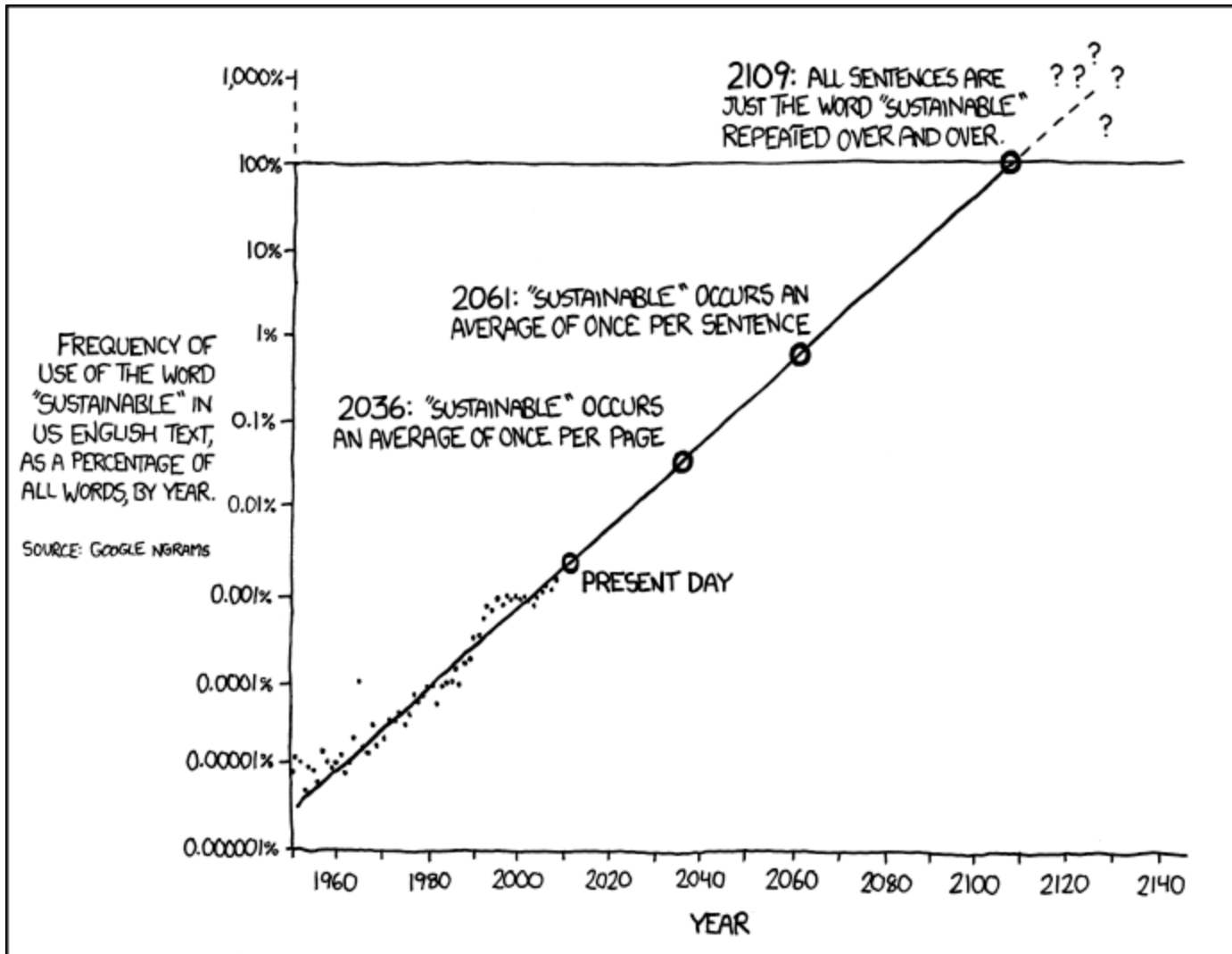
# Python Dictionaries

Function (All operate on Dictionaries)	Input	Output	Example
<code>keys()</code>	None	List of keys	<pre>&gt;&gt;&gt; freq2.keys() ['the', 'cat']</pre>
<code>values()</code>	None	List of values	<pre>&gt;&gt;&gt; freq2.values() [3, 2]</pre>
<code>has_key(&lt;key&gt;)</code>	Key	Boolean	<pre>&gt;&gt;&gt; freq2.has_key('missing') False</pre>
<code>&lt;key&gt; in &lt;dict&gt;</code>	(means same as above)		<pre>&gt;&gt;&gt; 'the' in freq2 True</pre>
<code>del(&lt;dict&gt;[&lt;key&gt;])</code>	Dict. Entry	None	<pre>&gt;&gt;&gt; del(freq2['cat'])</pre>

Do Task 1

- Keys Are Unique!
- Dictionaries are Unordered! Unlike lists, which are linear.

# Break



THE WORD "SUSTAINABLE" IS UNSUSTAINABLE.

<http://xkcd.com/1007/>

# The Big Picture

## Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

## Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

# Python Dictionaries

Function (All operate on Dictionaries)	Input	Output	Example
<code>keys()</code>	None	List of keys	<pre>&gt;&gt;&gt; freq2.keys() ['the', 'cat']</pre>
<code>values()</code>	None	List of values	<pre>&gt;&gt;&gt; freq2.values() [3, 2]</pre>
<code>has_key(&lt;key&gt;)</code>	Key	True or False	<pre>&gt;&gt;&gt; freq2.has_key('missing') False</pre>
<code>&lt;key&gt; in &lt;dict&gt;</code>	(means same as above)		<pre>&gt;&gt;&gt; 'the' in freq2 True</pre>
<code>del(&lt;dict&gt;[&lt;key&gt;])</code>	Dict. Entry	None	<pre>&gt;&gt;&gt; del(freq2['cat'])</pre>

- Keys Are Unique!
- Dictionaries are Unordered! Unlike lists, which are linear.

# The Big Picture

## Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

## Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

# Building a Concordance

The cat had a hat. The cat sat on the hat.  
0 1 2 3 4 5 6 7 8 9 10

Word	List of Positions	Frequency
the	[0, 5, 9]	3
cat	[1, 6]	2
had	[2]	1
a	[3]	1
hat	[4, 10]	2
sat	[7]	1
on	[8]	1

# The Big Picture

## Overall Goal

Build a Concordance of a text

- *Locations* of words
- *Frequency* of words

## Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

This is part of  
your HW

# Project 1

Go to the Projects page on the class website

- Make sure that your website shows up properly.
- Look through a few other people's projects