√ 4. Source: Files on Hand

✓ 3. Dataset: Finding the Best Movies

√ 5. Flat File Structure

✓ 1. Introduction

✓ 2. Lesson Outline

✓ 6. Flat Files in Python

√ 7. Source: Web Scraping

√ 8. HTML File Structure

✓ 9. HTML Files in Python

✓ 10. Flashforward 1

11. Source: Downloading Files from the Internet

✓ 12. Text File Structure

√ 13. Text Files in Python

14. Source: APIs (Application Programming Interfaces)

15. JSON File Structure

16. JSON Files in Python

17. Mashup: APIs, Downloading Files

Programmatically, JSON

18. Mashup Solution

19. Flashforward 2

20. Storing Data

21. Relational Database Structure

22. Relational Databases in Python

23. Other File Formats

24. You Can Iterate

25. Gathering Summary

26. Conclusion

27. Glossary

**Text Files** 

- Use a specific character set
- Contain no formatting, like italics or bolding
- Have no media, like images or video
- Lines of text are separated by newline characters or backslash end in Python that are typically invisible in most software applications
- Can be viewed and edited in simple text editors, which makes sense.

Flat files are text files that have a specific structure. The Roger Ebert review text files shown in the video are just blobs of text without a defined structure like the tabular structure in the TSV format.

We're going to focus on this no-structure, just a blob-of-text file. But t all of the techniques you we will learn can be done with any text file regardless of structure.

### **Encoding and Character Sets**

Have you ever opened a document and seen garbled characters? That happens when your text editor, browser, or word processor is assuming the wrong encoding. You simply need to select the right encoding to display the document properly.

Character sets and encodings are two things that every programmer needs to be aware of when working with any text data

- **Character sets** are the collections of characters that are available for use in a system.
- **Encoding** is the scheme for converting the character sets bits to letters and numbers.

Read the articles below to learn more.

# **Encodings and Character Sets Articles**

The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!) by Joel Spolsky

An excerpt:

The Single Most Important Fact About Encodings

If you completely forget everything I just explained, please remember one extremely important fact. It does not make sense to have a string without knowing what encoding it uses. You can no longer stick your head in the sand and pretend that "plain" text is ASCII.

There Ain't No Such Thing As Plain Text

If you have a string, in memory, in a file, or in an email message, you have to know what encoding it is in or you cannot interpret it or display it to users correctly.

Almost every stupid "my website looks like gibberish" or "she can't read my emails when I use accents" problem comes down to one naive programmer who didn't understand the simple fact that if you don't tell me whether a particular string is encoded using UTF-8 or ASCII or ISO 8859-1 (Latin 1) or Windows 1252 (Western European), you simply cannot display it correctly or even figure out where it ends. There are over a hundred encodings and above code point 127, all bets are off."

What Every Programmer Absolutely, Positively Needs To Know About Encodings And Character Sets To Work With Text

An article by Joel Spolsky entitled The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!) is a nice introduction to the topic and I greatly enjoy reading it every once in a while. I hesitate to refer people to it who have trouble understanding encoding problems though since, while entertaining, it is pretty light on actual technical details. I hope this article can shed some more light on what exactly an encoding is and just why all your text screws up when you least need it.

Any character can be encoded in many different bit sequences and any particular bit sequence can represent many different characters, depending on which encoding is used to read or write them. The reason is simply because different encodings use different numbers of bits per characters and different values to represent different characters."

# **Unicode and Python**

In Python 3, there is:

- one text type: str , which holds Unicode data and
- two byte types: bytes and bytearray

The Stack Overflow answers here explain the different use cases well.

# More Information

- If you're still confused about the difference between character sets and encoding, check out these articles:
  - The difference between UTF-8 and Unicode? More About Unicode in Python 2 and 3

# **New Terms**

	Term	Definition
	Character Set	A collection of characters that is available for use
	Encoding	The scheme for converting the character set bits into letters and numbers
	Text File	A file that uses specific character set where lines of text are separated by newline characters and contains no formatting or media

NEXT